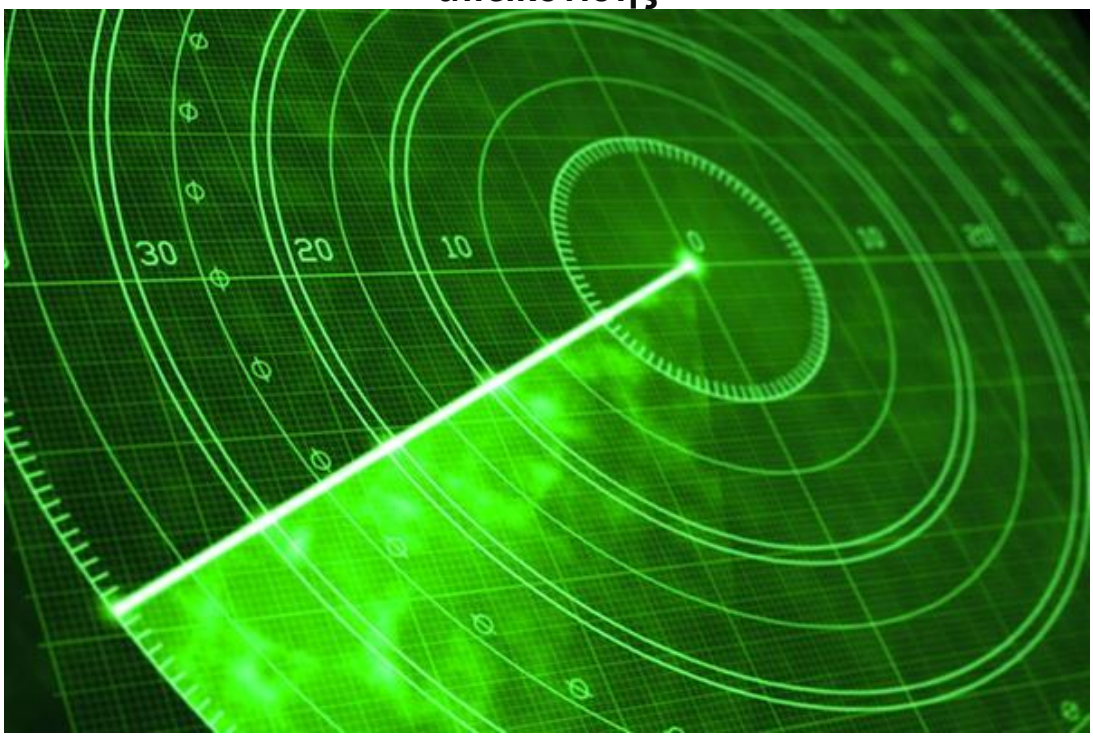
ΑΕΙ ΠΕΙΡΑΙΑ Τ.Τ.
ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ
ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΗΛΕΚΤΡΟΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ Τ.Ε.

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Δημιουργία Sonar με Arduino και κατασκευή του περιβάλλοντος απεικόνισης



Πλούταρχος Ι. Ξυναριανός

Εισηγητής: κ. Αναστασία Βελώνη, Λέκτορας Εφαρμογών

ΑΘΗΝΑ,ΙΟΥΝΙΟΣ 2018

(Κενό φύλλο)

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Δημιουργία Sonar με Arduino και κατασκευή του περιβάλλοντος απεικόνισης

Πλούταρχος Ι. Ξυναριανός
Α.Μ. 43209

Εισηγητής:

κ. Αναστασία Βελώνη, Λέκτορας Εφαρμογών

Εξεταστική Επιτροπή:

Ημερομηνία εξέτασης

(Κενό φύλλο)

ΕΥΧΑΡΙΣΤΙΕΣ

Θα ήθελα να εκφράσω τις ευχαριστίες μου στην κ. Αναστασία Βελώνη για την δυνατότητα που μου έδωσε να πραγματοποιήσω την πτυχιακή μου εργασία. Οι σημαντικές υποδείξεις και συμβουλές της, με κατεύθυναν σ' ένα σωστό τρόπο σκέψης, έκφρασης και κυρίως αποτύπωσηςτου τεχνικού μέρους της πτυχιακής μου αναφοράς.

Υποχρέωση μου να ευχαριστήσω ακόμα και όλους του καθηγητές του τμήματος Μηχανικών Ηλεκτρονικών Υπολογιστικών Συστημάτων για τις πολύτιμες γνώσεις που μου προσέφεραν όλα αυτά τα χρόνια.

Τέλος, θέλω να εκφράσω ένα τεράστιο ευχαριστώ στην οικογένεια μου, τους φίλους μου και όλους τους κοντινούς μου ανθρώπους για την στήριξη και την εμπιστοσύνη που μου έδειξαν όλα αυτά τα χρόνια των σπουδών μου.

(Κενό φύλλο)

ΠΕΡΙΛΗΨΗ

Η παρούσα πτυχιακή εργασία ασχολείται με την δημιουργία κυκλώματος Σόναρ(ηχοεντοπιστικό σύστημα ανίχνευσης αντικειμένων) με τον μικροελεγκτή μονής πλακέταςArduinoMega 2560 και την ανάπτυξη γραφικού περιβάλλοντος απεικόνισης παρόμοιο με αυτό το οποίο χρησιμοποιούν τα πλοία και τα υποβρύχια. Η εργασία θα περιέχει οπτικοακουστικό υλικό σε κάθε μέρος τηςυλοποίησής της , όπως η δημιουργία του κυκλώματος βήμα-βήμα, ο προγραμματισμός του Arduino στην Wiring γλώσσα μέσω του ArduinoIDE, η δημιουργία του γραφικού περιβάλλοντος μέσω του ProcessingIDE, η δημιουργία σύνδεσης μέσω των ArduinoIDE και Processing, η ανάλυση του κάθε εξαρτήματος που θα χρησιμοποιηθεί καθώς και η λειτουργία των συστημάτων Σόναρ.Ακόμη θα παρουσιαστεί η ιστορική αναδρομή του κάθε εξαρτήματος και προγράμματος/ πλατφόρμας που χρησιμοποιήθηκε όπως και οι περαιτέρω δυνατότητες του. Όλα τα παραπάνω μέρη της οπτικοακουστικής παρουσίασης της εργασίας θα αναλυθούν περαιτέρω στην φυσική, γραπτή της μορφή στην οποία θα παρουσιαστούν λεπτομερώς όλες οι πληροφορίες και τα συμπεράσματα που προέκυψαν μέσω της εκποίησης αυτής της εργασίας.

ABSTRACT

This diploma thesis deals with the creation of a Sonar circuit with the Arduino Mega 2560 single-board microcontroller and the development of a graphical display environment similar to that used by ships and submarines. The thesis will contain a multimedia presentation in every part of its implementation, such as creating step-by-step the circuit, programming Arduino in Wiring language through Arduino IDE, creating the graphic environment through the Processing IDE, creating a connection through Arduino IDE and Processing, the analysis of each component that is going to be used, and the operation of Sonar systems. It will also show the historical overview of each component and program/platform used as well as its further capabilities. All the above parts of the multimedia presentation of the thesis will be further analyzed in its natural, written form in which will be presented in detail all the information and conclusions that have resulted from the implementation of this thesis

ΕΠΙΣΤΗΜΟΝΙΚΗ ΠΕΡΙΟΧΗ:

ΣΥΝΤΟΜΟΓΡΑΦΙΕΣ : ολοκληρωμένο περιβάλλον ανάπτυξης(IDE) , ανέβασμα κώδικα (upload) , Γραφικό περιβάλλον (GUI)

ΛΕΞΕΙΣΚΛΕΙΔΙΑ: Sonar, Arduino, Processing, IDE, HC-SR04, Servo motor, Ultrasonic, υπέρηχοι

ΠΕΡΙΕΧΟΜΕΝΑ

Πίνακας περιεχομένων

Πίνακας Εικόνων	
Εικόνα 1.1 Σόναρ κατά LeonardoDaVinci	Σελίδα 14
Εικόνα 1.2 Επαφή του ταλαντωτή με την θάλασσα από τον Reginald Fessenden το 1914	Σελίδα 15
Εικόνα 1.3 ASDIC συσκευή απεικόνισης γύρω στο 1944	Σελίδα 16
Εικόνα 1.4 Υποβρύχια κλάσης T (RoyalNavy) WW2	Σελίδα 17
Εικόνα 1.5 Fundamentals of Sonar Αρχικήέκδοση 1957	Σελίδα 19
Εικόνα 1.6 Tonpliz *piezotransducer voltage distribution*	Σελίδα20
Εικόνα 1.7 Αρχή λειτουργίας ενεργητικού Σόναρ	Σελίδα21
Εικόνα 1.8 Συστήματα Σόναρ, πλεονεκτήματα και μειονεκτήματα	Σελίδα 22
Εικόνα 1.9 ATmega168 μικροελεγκτής	Σελίδα 23
Εικόνα 1.10Arduino RS232	Σελίδα 25
Εικόνα 1.11 Arduino Diecimila	Σελίδα 25
Εικόνα 1.12 Arduino Duemilanove	Σελίδα 25
Εικόνα 1.12 Arduino Uno R2	Σελίδα 25
Εικόνα 1.13 Arduino Leonardo	Σελίδα 26
Εικόνα 1.14 Arduino Pro	Σελίδα 26
Εικόνα 1.15 Arduino Mega 2560	Σελίδα 26
Εικόνα 1.16 Arduino Nano	Σελίδα 26
Εικόνα 1.17 Arduino LilyPad	Σελίδα 26
Εικόνα 1.18 Arduino Robot	Σελίδα 27
Εικόνα 1.19 Arduino Esplora	Σελίδα 27
Εικόνα 1.20 Arduino Ethernet	Σελίδα 27
Εικόνα 1.21 Arduino Yun	Σελίδα 27
Εικόνα 1.22 Arduino Due	Σελίδα 27
Εικόνα 1.23 Βασικά στοιχεία ArduinoMega 2560	Σελίδα 28
Εικόνα 1.24 Arduino Mega 2560 Pin Mapping	Σελίδα30
Εικόνα 2.1Αισθητήρας HC-SR04	Σελίδα 32
Εικόνα 2.2 Διάκριση Ηχητικών Κυμάτων σύμφωνα με την συχνότητα	Σελίδα 32

1.ΕΙΣΑΓΩΓΗ ΣΤΑ SONAR

Τα ηχοεντοπιστικά συστήματα Sonar, λέξη η οποία αποτελεί ακρώνυμο των **SO**und **N**avigation **A**nd **R**anging είναι η πρακτική εφαρμογή μίας τεχνικής που χρησιμοποιεί τη διάδοση ήχου (συνήθως υποβρύχια, όπως και στην υποβρύχια πλοήγηση) για πλοήγηση, επικοινωνία ή ανίχνευση αντικειμένων πάνω ή κάτω από την επιφάνεια του νερού. Ο σκοπός των συστημάτων Σόναρ είναι ο εντοπισμός/ανίχνευση, αναγνώριση/ταξινόμηση και παρακολούθηση υποβρυχίων σκαφών και διαφόρων αντικειμένων, η ακουστική χαρτογράφηση/τομογραφία του βυθού, η ναυτιλία πλοίων επιφανείας και υποβρυχίων καθώς επίσης οι υποθαλάσσιες επικοινωνίες & τηλεμετρία

Τα Σόναρ χωρίζονται σε 2 κατηγορίες:

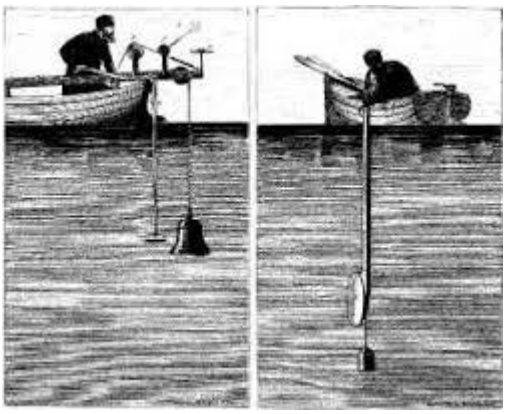
Τα παθητικά Σόναρ τα οποία ουσιαστικά ακούνε τους ήχους που παράγουν τα άλλα σκάφη και τα ενεργητικά Σόναρ τα οποία εκπέμπουν ηχητικούς παλμούς και ακούνε την ηχώ τους. Το Σόναρ μπορεί να χρησιμοποιηθεί ως μέσο ακουστικής θέσης και μέτρησης των χαρακτηριστικών ηχούς των "στόχων" στο νερό. Στόχος είναι το κάθε αντικείμενο που μπορεί να ανιχνευτεί μέσω των συστημάτων Σόναρ.

Η ακουστική θέση στον αέρα χρησιμοποιήθηκε πολύ πριν από την εισαγωγή των συστημάτων ραντάρ. Τα Σόναρ μπορούν να χρησιμοποιηθούν στον αέρα για ρομποτική πλοήγηση και τα συστήματα SODAR (**SO**nic **D**etection **A**nd **R**anging)για τη μέτρηση της ταχύτητας του ανέμου σε διάφορα ύψη πάνω από το έδαφος και τη θερμοδυναμική δομή του κάτω στρώματος της ατμόσφαιρας.Ο όρος Σόναρ χρησιμοποιείται επίσης για τον εξοπλισμό που χρησιμοποιείται για την παραγωγή και λήψη του ήχου.

Οι ακουστικές συχνότητες που χρησιμοποιούνται στα συστήματα σόναρ ποικίλλουν από πολύ χαμηλές (υποηχητικές) έως εξαιρετικά υψηλές (υπερηχητικές).Η μελέτη του υποβρύχιου ήχου είναι γνωστή ως υποβρύχια ακουστική ή υδροακουστική.

1.1 Ιστορική αναδρομή συστημάτων Sonar

Παρότι ορισμένα ζώα (δελφίνια και νυχτερίδες) έχουν χρησιμοποιήσει ήχο για επικοινωνία και ανίχνευση αντικειμένων για εκατομμύρια χρόνια, η χρήση τους από τον άνθρωπο στο νερό καταγράφεται για πρώτη φορά από τον Leonardo Da Vinci το 1490 με τον εξής τρόπο: Ένας σωλήνας που έχει εισαχθεί στο νερό λέγεται ότι χρησιμοποιείται για την ανίχνευση πλοίωνμε την τοποθέτηση του αυτιού στην άκρη του σωλήνα.

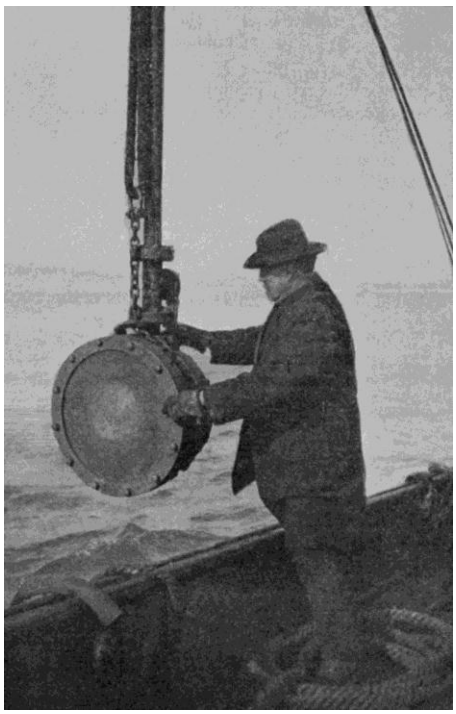


Εικόνα 1.1 Σόναρ κατά LeonardoDaVinci

Τον 19^{ο} αιώνα χρησιμοποιήθηκε ένα υποβρύχιο κουδούνι στους φάρους των πλοίων για να προειδοποιεί τα πλοία για τυχών κινδύνους. Η χρήση του ήχου για την υποβρύχια “ήχο-πλοήγηση” όπως η νυχτερίδες χρησιμοποιούν τον ήχο για την εναέρια πλοήγηση τους φαίνεται να προκλήθηκε από την καταστροφή του Τιτανικού το 1912.Το πρώτο δίπλωμα ευρεσιτεχνίας στον κόσμο για μια υποβρύχια ηχοακουστική συσκευή καταχωρήθηκε στο βρετανικό γραφείο διπλωμάτων ευρεσιτεχνίας από τον αγγλικό μετεωρολόγο Lewis Fry Richardson ένα μήνα μετά το ναυάγιο του Τιτανικού [2] και ένας Γερμανός φυσικός ονομαζόμενος Alexander Behm απέκτησε δίπλωμα ευρεσιτεχνίας ηχοακουστικής συσκευής το 1913.

Ο καναδός μηχανικός Reginald Fessenden, ενώ εργαζόταν για την Submarine Signal Company στη Βοστώνη, δημιούργησε ένα πειραματικό σύστημα που ξεκίνησε το 1912, και στην συνέχεια δοκιμάστηκε στο λιμάνι της Βοστώνης και τελικά το 1914 από το US Revenue (σημερινή Coast Guard) cutter Maimi στο Grand Banks, του Newfoundland Canada. Σε αυτήν την επίδειξη ο Fessedenέδειξε την ηχητική βάθους (για προσδιορισμό του βάθους την

θάλασσας), υποθαλάσσιες επικοινωνίες (κώδικας Morse) και ήχο-πλοήγηση (εντοπίζοντας ένα παγόβουνο σε απόσταση 3,2 χιλιομέτρων(2 μίλια).Ο [5] Ο αποκαλούμενος ταλαντωτής Fessenden ο οποίος λειτουργούσε με συχνότητα 540 Hz το 1915 δοκιμάστηκε ακόμη και στα 100 kHz.Τα μοντέλα ταλαντωτή Fessenden (540, 1000 και 3000 Hz) ήταν τόσο επιτυχημένα που χρησιμοποιήθηκαν μέχρι και κατά τη διάρκεια του Β Παγκοσμίου Πολέμου για σκοπούς σόναρ και ανίχνευσης ναρκών. Τα δέκα Βρετανικά υποβρύχια Η κλάσης που φτιάχτηκαν στο Montreal το 1915 ήταν εξοπλισμένα με έναν ταλαντωτή Fessenden. Παρά τα σπουδαία επιτεύγματα των ταλαντωτών αυτών κανενός η ύπαρξη δεν είναι γνωστή ότι υφίσταται σήμερακαι δεν έχουν πραγματοποιηθεί ποτέ σύγχρονες ακουστικές μετρήσεις για τον καθορισμό της ακουστικής τους απόδοσης.

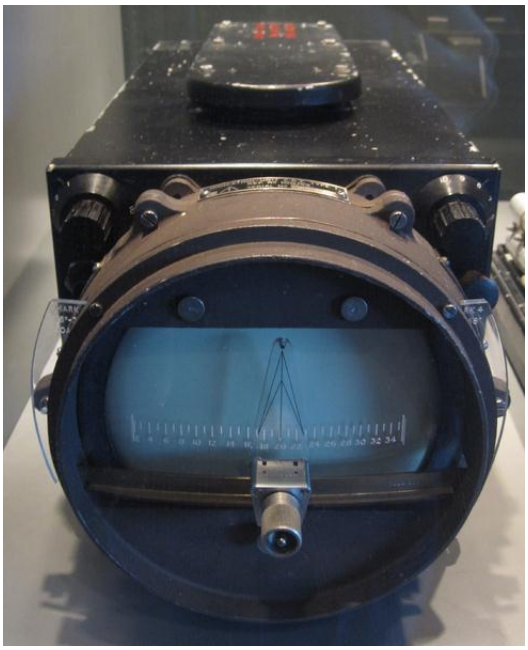


Εικόνα 1.2 Επαφή του ταλαντωτή με την θάλασσα από τον Reginald Fessenden το 1914

Κατά τη διάρκεια του Α 'Παγκοσμίου Πολέμου, η ανάγκη εντοπισμού υποβρυχίων οδήγησε σε περισσότερες έρευνες σχετικά με τη χρήση του ήχου. Οι Βρετανοί χρησιμοποίησαν νωρίτερα συσκευές υποβρύχιας ακρόασης που ονομάζονταν **υδρόφωνα**, ενώ ο Γάλλος φυσικός Paul Langevin, συνεργαζόμενος με τον Ρώσο μετανάστη ηλεκτρολόγο μηχανικό Constantin Chilowsky, εργάστηκε για την ανάπτυξη ενεργών ηχητικών συσκευών για την ανίχνευση υποβρυχίων το 1915. Παρόλο που οι πιεζοηλεκτρικοί και μαγνητοσυστολικοί μετατροπείς αργότερα υπερίσχυσαν από τους ηλεκτροστατικούς μετατροπείς που χρησιμοποίησαν η δουλεία του ενέπνευσε πολλά μελλοντικά σχέδια.Οι ελαφρές και ευαίσθητες στο ήχο πλαστικές μεμβράνες και οι οπτικές ίνες έχουν χρησιμοποιηθεί για υδρόφωνα (ακουστικό-ηλεκτρικοί μετατροπείς για χρήση σε νερό), ενώ έχουν αναπτυχθεί προβολείς τεχνολογίας Terfenol-D και PMN (νιοβικό μαγνήσιο μόλυβδου).

ASDIC

Το 1916, στο πλαίσιο του Βρετανικού Συμβουλίου Έρευνας και Έρευνας, ο καναδός φυσικός Robert William Boyle ανέλαβε το ενεργό πρόγραμμα ανίχνευσης ήχου με τον A. B. Wood, με σκοπό να παράγει ένα πρωτότυπο για δοκιμές στα μέσα του 1917. Το έργο αυτό για το αντι-υποβρηχιακό τμήμα του Βρετανικού Ναυτικούπραγματοποιήθηκε με απόλυτη μυστικότητα και χρησιμοποίησε πιεζοηλεκτρικούς κρυστάλλους χαλαζία για να παράγει την πρώτη πρακτική συσκευή ανίχνευσης ήχου υποβρύχιας τεχνολογίας στον κόσμο.Για να διατηρηθεί η μυστικότητα, δεν έγινε καμία αναφορά στον πειραματισμό του ήχου ή στον χαλαζία - η λέξη που χρησιμοποιήθηκε για να περιγράψει την πρώιμη εργασία ("supersonics") άλλαξε σε “ASD”ics και το υλικό χαλαζία σε "ASD"ivite . Τα "ASD" παραπέμπουν στα αρχικά του "Anti-Submarine Division"(Αντί Υποβρυχιακό Τμήμα), εξ ου και το βρετανικό ακρώνυμο ASDIC.Το 1939, απαντώντας σε μια ερώτηση από το αγγλικό λεξικό της Οξφόρδης σε τι παραπέμπουν τα αρχικά αυτά, το Ναυαρχείο ανέπτυξε την ιστορία ότι υπήρχε για την «Allied Submarine Detection Investigation Committee» (Επιτροπή Διερεύνησης Ανίχνευσης Υποβρυχίων) κάτι που ακόμη και σήμερα εξακολουθεί να πιστεύεται, αν και καμία επιτροπή που φέρει αυτό το όνομα δεν έχει υπάρχει στα αρχεία του Ναυαρχείου που βρέθηκαν.

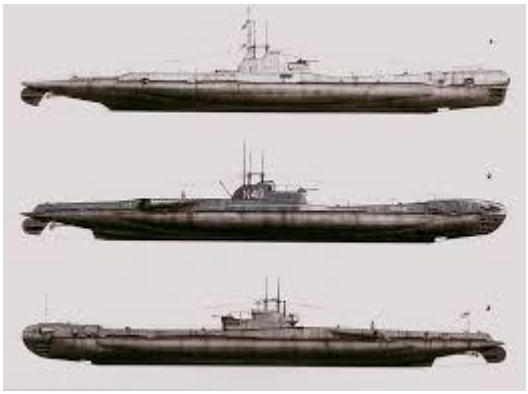


Εικόνα 1.3 ASDIC συσκευή απεικόνισης γύρω στο 1944

Μέχρι το 1918, τόσο η Γαλλία όσο και η Βρετανία είχαν κατασκευάσει παρόμοια πρωτότυπα ενεργά συστήματα. Οι Βρετανοί δοκίμασαν το ASDIC τους στο HMS Antrim το 1920 και άρχισαν την παραγωγή το 1922. Η 6^{η}αρμάδα καταστροφέων είχε σκάφη εξοπλισμένα με ASDIC το 1923. Στο Πόρτλαντ το 1924 ιδρύθηκε ένα σχολείο κατά των υποβρυχίων HMS Osprey μαζί με ένα εκπαιδευτικό σχήμα τεσσάρων πλοίων. Το αμερικανικό Sonar QB εμφανίστηκε στο προσκήνιο το 1931.

Από το ξέσπασμα του Β' Παγκόσμιου Πολέμου, το Βρετανικό Ναυτικό (RoyalNavy) είχε πέντε διαφορετικές κατηγορίες πλοίων και διαφορετικές για υποβρύχια τα οποία ήταν ενσωματωμένα σε ένα πλήρες αντι-υποβρυχιακό σύστημα.

Η αποτελεσματικότητα της πρώιμης τεχνολογίας ASDIC εμποδίστηκε από την χρήση της βόμβας βάθους με σκοπό την καταστροφή υποβρυχίων. Επρόκειτο για ένα δοχείο σε σχήμα βαρελιού γεμάτο εκρηκτικά και εφοδιασμένο μεέναν πυροσωλήνα ρυθμισμένο ώστε να προκαλέσει έκρηξη σε προκαθορισμένο βάθος. Η βόμβα ριπτόταν είτε από ειδικό καταπέλτη βομβοβόλο είτε κυλιόμενη από ειδική εξέδρα, μονή ή διπλή που φέρονταν στη πρύμνη των πλοίων. Στόχος της βόμβας βάθους ήταν να προκαλέσει οποιαδήποτε ζημιά που θα έβγαζε εκτός ενέργειας το εντοπισμένο προηγουμένως αντίπαλο υποβρύχιο, μέχρι ακόμη και τη βύθισή του αν αυτό δεν αναγκαζόταν σε ανάδυση. Αυτό όμως απαιτούσε το πλοίο να περάσει πάνω από το στόχο του πριν προβεί σε επίθεση και ως αποτέλεσμα χανόταν η επαφή του ASDIC την στιγμή της επίθεσης. Τις κρίσιμες στιγμές αυτές ο αρχηγός του υποβρυχίου μπορούσε να αποφύγει την επίθεση.



Εικόνα 1.4 Υποβρύχια κλάσης T (RoyalNavy) WW2

Η κατάσταση αυτή βελτιώθηκε με την συνεργασίας πολλών πλοίων μαζί και με την μέθοδο όπλων "προ" πορείας όπως οι "Hedgehog" και "Squid" οπού εμβόλιζαν βόμβες "προ" πορείας στον στόχο τους, διατηρώντας την ASDIC επαφή. Η συνεχής ανάπτυξη των Βρετανικών συστημάτων ASDIC τα οποία χρησιμοποιούσαν διαφορετικών διαμέτρων ακτίνες συντέλεσε στην συνεχής εξάλειψη "τυφλών" σημείων ενώ αργότερα χρησιμοποιήθηκαν ακουστικές τορπίλες.

Κατά την έναρξη του Β 'Παγκοσμίου Πολέμου, η βρετανική τεχνολογία ASDIC μεταφέρθηκε στις Ηνωμένες Πολιτείες. Η έρευνα για τον ASDIC και τον υποβρύχιο ήχο επεκτάθηκε στο Ηνωμένο Βασίλειο και στις ΗΠΑ. Πολλοί νέοι τύποι στρατιωτικού εντοπισμού ήχου αναπτύχθηκαν. Αυτά περιλάμβαναν τα sonobuoys, που αναπτύχθηκαν για πρώτη φορά από τους Βρετανούς το 1944 με την κωδική ονομασία High Tea (dipping/dunking) τύπος σόναρ και σόναρ εντοπισμού ναρκώνΑυτό είναι το έργο το οποίο αποτέλεσε τη βάση για μεταπολεμικές εξελίξεις σχετικά με την αντιμετώπιση των πυρηνικών υποβρυχίων.

SONAR

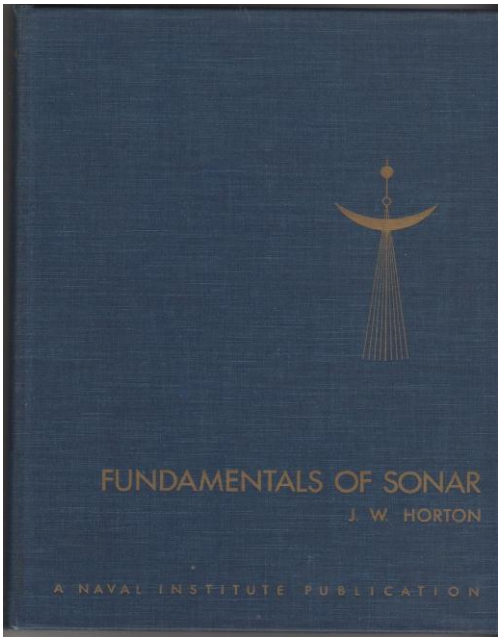
Κατά τη διάρκεια της δεκαετίας του 1930 οι Αμερικανοί μηχανικοί ανέπτυξαν τη δική τους υποβρύχια τεχνολογία ανίχνευσης ήχου όπου και έγιναν σημαντικές ανακαλύψεις, όπως οι θερμοκλείδες, που θα βοηθούσαν στη μελλοντική ανάπτυξη τον υποβρύχιων ακουστικών συστημάτων. Μετά από ανταλλαγή τεχνικών πληροφοριών κατά τη διάρκεια του Δευτέρου Παγκοσμίου Πολέμου, οι Αμερικανοί άρχισαν να χρησιμοποιούν τον όρο SONAR για τα συστήματά τους, που αποτέλεσε ισοδύναμο με τον όρο του RADAR.

Εργαστήριο υποβρύχιου ήχου του Πολεμικού Ναυτικού των ΗΠΑ

Το 1917, το Πολεμικό Ναυτικό των Ηνωμένων Πολιτειών απέκτησε για πρώτη φορά τις υπηρεσίες του Dr. J. Warren Horton. Σε άδεια από την Bell Labs, υπηρέτησε στην κυβέρνηση ως τεχνικός εμπειρογνώμονας, πρώτα στον πειραματικό σταθμό στο Nahant, στη Μασαχουσέτη και αργότερα στο αμερικανικό ναυτικό αρχηγείο στο Λονδίνο. Στο Nahant εφάρμοσε τον πρόσφατα ανεπτυγμένο σωλήνα κενού, που στη συνέχεια συνδέθηκε με τις φάσεις σχηματισμού του πεδίου εφαρμοσμένης επιστήμης, γνωστής τώρα ως ηλεκτρονική, για την ανίχνευση υποβρυχίων σημάτων. Ως αποτέλεσμα, το μικρόφωνο του κουμπιού άνθρακα, το οποίο είχε χρησιμοποιηθεί σε προηγούμενο εξοπλισμό ανίχνευσης, αντικαταστάθηκε από το σύγχρονο υδρόφωνο. Επίσης κατά τη διάρκεια αυτής της περιόδου, πειραματίστηκε με μεθόδους συρόμενης ανίχνευσης. Αυτό οφείλεται στην αυξημένη ευαισθησία της συσκευής του. Οι αρχές εξακολουθούν να χρησιμοποιούνται στα σύγχρονα συρόμενα συστήματα σόναρ.

Για να ανταποκριθεί στις αμυντικές ανάγκες της Μεγάλης Βρετανίας, ο Dr.Horton αποστέλλεται στην Αγγλία για να εγκαταστήσει στα βυθισμένα υδρόφωνα της Θάλασσας της Ιρλανδίας συνδεδεμένα με ένα στύλο ακρόασης στην ακτή με υποβρύχιο καλώδιο. Ενώ αυτός ο εξοπλισμός φορτώθηκε στο σκάφος που θα τοποθετούσε τα καλώδια, ο πόλεμος τελείωσε και ο Dr. Horton επέστρεψε σπίτι.

Κατά τη διάρκεια του Β 'Παγκοσμίου Πολέμου, συνέχισε τη συμμετοχή του στην ανάπτυξη συστημάτων σόναρ για την ανίχνευση υποβρυχίων, ναρκών και τορπιλών. Δημοσίευσε το βιβλίο FundamentalsofSonar(Βασικά στοιχεία του Σόναρ) το 1957 ως Διευθύνων Σύμβουλος Ερευνών στο Εργαστήριο Υποβρύχιου Ήχου του Πολεμικού Ναυτικού των ΗΠΑ. Κατείχε αυτή τη θέση μέχρι το 1959, όταν έγινε Τεχνικός Διευθυντής, θέση που κατείχε μέχρι την υποχρεωτική συνταξιοδότηση το 1963.



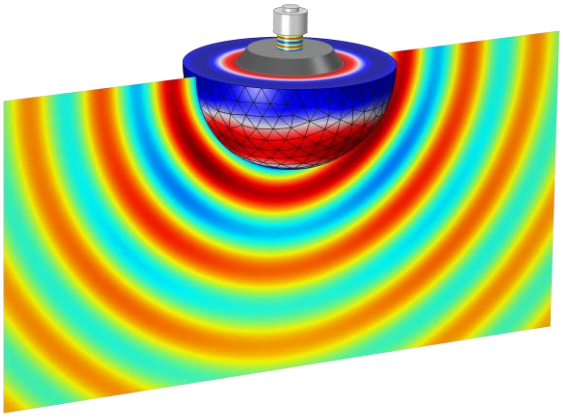
Εικόνα 1.5 Fundamentals of Sonar
Αρχικήέκδοση 1957

ActiveSonar (Ενεργητικό Σόναρ)

Το ενεργητικό Σόναρ χρησιμοποιεί ένα ηχητικό πομπό και έναν ηχητικό δέκτη. Όταν ο πομπός και ο δέκτης βρίσκονται στην ίδια θέση έχουμε μονοστατική λειτουργία. Όταν ο πομπός και ο δέκτη είναι ξεχωριστά έχουμε διστατική λειτουργία. Όταν χρησιμοποιηθούν περισσότεροι πομποί ή περισσότεροι δέκτες τότε η λειτουργία είναι πολυστατική. Τα περισσότερα σόναρ χρησιμοποιούνται μονοστατικά με την ίδια συστοιχία συχνά να χρησιμοποιείται για μετάδοση και λήψη.Τα ενεργά πεδία sonobuoy έχουν την δυνατότητα να λειτουργούν πολυστατικά.

Το ενεργό σόναρ δημιουργεί παλμό ήχου, συχνά αποκαλούμενο "**ping**", και στη συνέχεια ακούει την αντανάκλαση “ηχώ-**echo**” του παλμού.Αυτός ο ηχητικός παλμός δημιουργείται ηλεκτρονικά χρησιμοποιώντας έναν προβολέα σόναρ που αποτελείται από μια γεννήτρια σημάτων, έναν ενισχυτή ισχύος και μια συστοιχία ηλεκτροακουστικών μετατροπέων.

Ένας φορέας δέσμης χρησιμοποιείται συνήθως για να συγκεντρώσει την ακουστική ισχύ σε μια ακτίνα, η οποία μπορεί να σαρωθεί για να καλύψει τις απαιτούμενες γωνίες αναζήτησης.Γενικά, οι ηλεκτροακουστικοί μετατροπείς είναι τύπου Tonpilz και ο σχεδιασμός τους μπορεί να βελτιστοποιηθούν έτσι ώστε να επιτυγχάνεται μέγιστη απόδοση σε όλο το εύρος ζώνης, προκειμένου να βελτιστοποιηθεί η απόδοση του συνολικού συστήματος.



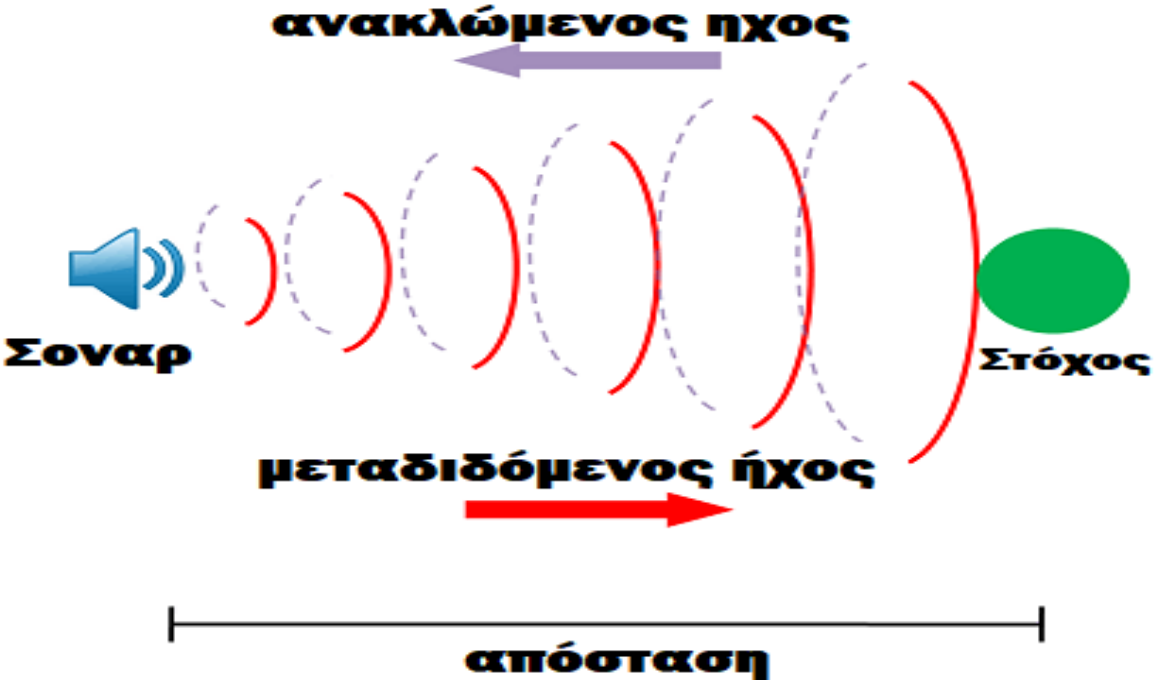
Εικόνα 1.6 Tonpliz *piezotransducer voltage distribution*

Περιστασιακά, ο ακουστικός παλμός μπορεί να δημιουργηθεί με άλλα μέσα, π.χ.

- χημική χρήση εκρηκτικών
- αεροβόλα όπλα
- πηγές ήχου πλάσματος

Για να μετρηθεί η απόσταση ενός αντικειμένου(στόχου) μετριέται ο χρόνος από την μετάδοση ενός παλμού μέχρι τη λήψη της ηχώς του , πολλαπλασιάζεται με την ταχύτητα του ήχου και διαιρείται δια 2. Το σήμα του στόχου διαβάζεται μαζί με τον θόρυβο έπειτα από διαφόρων μορφών επεξεργασίας σημάτων που στα πιο απλά σόναρ μπορεί να είναι απλά μια μέτρηση ενέργειας. Μετά το σήμα παρουσιάζεται σε μια προκαθορισμένα από εμάς συσκευή λήψης στις οποίας την έξοδο μπορεί να προβληθεί το σήμα ή και ο θόρυβος. Αυτή η συσκευή λήψης μπορεί να έχει ως έξοδο μία απλή οθόνη , ένα ηχείο ή στα πιο εξελιγμένα σόναρ να πραγματοποιείται απεικόνιση μέσω λογισμικού.

$$\frac{\boldsymbol{Χρόνος\ που\ χρειαστηκε\ για\ αποστολή\ και\ λήψη\ της\ ηχούς\ των\ ηχητικών\ κυμάτων}}{\boldsymbol{2}}\ \boldsymbol{X\ \ Ταχύτητα\ \ Ηχου}$$

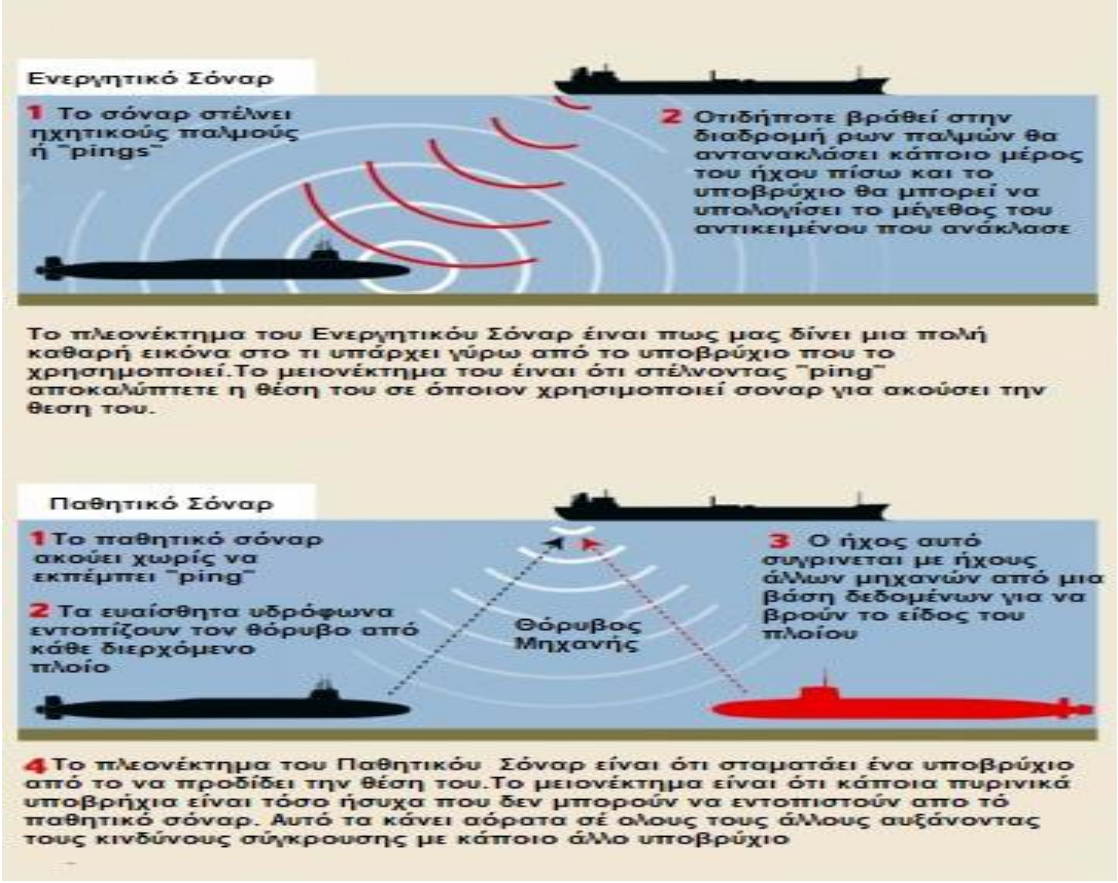


Εικόνα 1.7Αρχή λειτουργίας ενεργητικού Σόναρ

Επίσης το ενεργητικό σόναρ χρησιμοποιείται για την μέτρηση απόστασης μέσω νερού μεταξύ δύο σόναρ ή ένα συνδυασμό υδρόφωνων (υποβρύχια μικρόφωνα) και προβολείς (υποβρύχια ηχεία). Ένα σόναρ μεταδίδει και λαμβάνει ακουστικά σήματα “pings” . Όταν ένα σόναρ λαμβάνει ένα συγκεκριμένο σήμα ανταπόκρισης, αποκρίνεται μεταδίδοντας ένα συγκεκριμένο σήμα απάντησης.Όταν αυτή η τεχνική, χρησιμοποιείται με πολλαπλούς σόναρ / υδρόφωνα / προβολείς, μπορεί να υπολογίσει τις σχετικές θέσεις στατικών και κινούμενων αντικειμένων στο νερό.

PassiveSonar (Παθητικό Σόναρ)

Το παθητικό σόναρ χρησιμοποιείται για την ανίχνευση στόχων οι οποίοι παράγουν οι οποίοι παράγου ηχητικά κύματα είτε αυτά προέρχονται από τα σόναρ είτε από τον θόρυβο των μηχανών ενός υποβρυχίου ή πλοίου. Το παθητικό σόναρ ακούει τα ηχητικά κύματα που παράγουν χωρίς να εκπέμπει αυτό κάποιο ηχητικό κύμα. Η στρατιωτική χρήση είναι η κύρια χρήση των παθητικών σόναρ καθώς μπορούν να δώσουν μια σαφείς εικόνα για το τι υπάρχει γύρω τους χωρίς να προδώσουν την θέση τους.



Εικόνα 1.8 Συστήματα Σόναρ, πλεονεκτήματα και μειονεκτήματα

1.2 Ιστορική αναδρομή Arduino

Το Arduino είναι ένας μικροελεγκτής μονής πλακέτας, με λίγα λόγια μια απλή μητρική πλακέτα ανοικτού κώδικα με ενσωματωμένο μικροελεγκτή και εισόδους/εξόδους, η οποία μπορεί να προγραμματιστεί με τη γλώσσα Wiring . Το Arduino μπορεί να χρησιμοποιηθεί για την ανάπτυξη ανεξάρτητων διαδραστικών αντικειμένων αλλά και να συνδεθεί με υπολογιστή μέσω προγραμμάτων σε Processing, Max/MSP, Pure Data, SuperCollider. Οι περισσότερες εκδόσεις του Arduino μπορούν να αγοραστούν προ-συναρμολογημένες ενώ το διάγραμμα και πληροφορίες για το υλικό είναι ελεύθερα διαθέσιμα για αυτούς που θέλουν να συναρμολογήσουν το Arduino μόνοι τους.

Το project του Arduino ξεκίνησε στο Ινστιτούτο Σχεδίασης Αλληλεπίδρασης Ivrea (IDII) στην πόλη Ivrea της Ιταλίας. Την εποχή αυτή, οι μαθητές χρησιμοποίησαν έναν μικροελεγκτή BASIC Stamp ο οποίο κόστιζε γύρω στα 100€, κόστος το οποίο ήταν αρκετά σημαντικό για πολλούς φοιτητές.Το 2003, ο Hernando Barragán δημιούργησε την πλατφόρμα Wiring ως μέρος της διπλωματικής εργασίας στο IDII, υπό την επίβλεψη των Massimo Banzi και Casey Reas, οι οποίοι είναι γνωστοί από την εργασίας τους πάνω στην γλώσσα Processing.Ο στόχος του έργου ήταν η δημιουργία απλών, χαμηλού κόστους εργαλείων για τη δημιουργία ψηφιακών project από όχι ανθρώπου με λιγότερες γνώσεις από μηχανικούς.

Η πλατφόρμα Wiring αποτελούταν από μια τυπωμένη ηλεκτρονική πλακέτα (PCB)* με έναν ATmega168 μικροελεγκτή και ένα ολοκληρωμένο περιβάλλον ανάπτυξης (IDE) βασισμένο σε Processing και βιβλιοθήκες λειτουργιών για του εύκολο προγραμματισμό του μικροελεγκτή.Το 2003, ο Massimo Banzi, με τον David Mellis, έναν άλλο φοιτητή από το IDII και τον David Cuartielles, βοήθησαν στην υποστήριξη του φτηνότερου μικροελεγκτή ATmega8 πάνω στην πλακέτα Wiring. Αλλά αντί να συνεχίσουν να δουλεύουν πάνω στην Wiring, διέκοψαν το project και το ονόμασαν **Arduino**.

Η αρχική ομάδα του project **Arduino** αποτελούταν από τον Massimo Banzi, τον David Cuartielles, τον Tom Igoe, τον Gianluca Martino και τον David Mellis. Μετά την ολοκλήρωση της πλατφόρμας Wiring, διανεμήθηκαν ελαφρύτερες και φθηνότερες εκδόσεις στην κοινότητα ανοιχτού λογισμικού.

Εικόνα 1.9 ATmega168 μικροελεγκτής

Η AdafruitIndustries, προμηθευτής πλακετών και εξαρτημάτων Arduino στην Νέα Υόρκη, εκτίμησε πως πάνω από 300,000 πλακέτες Arduino έχουν βγει στην παραγωγή και το 2013 πάνω από 700,00 ήταν στα χέρια χρηστών τους.

Τον Οκτώβριο του 2016, ο Federico Musto, πρώην Διευθύνων Σύμβουλος του Arduino, εξασφάλισε στην ιδιοκτησία του το 50% της εταιρείας.

Σάλος προκαλέστηκε τον Απρίλιο του 2017 καθώς ο Wired ανέφερε πως ο Musto είχε πλαστογραφήσει την ακαδημαϊκή του πορεία καθώς στην ιστοσελίδα της εταιρίας του, στους προσωπικούς λογαριασμούς στο LinkedIn, ακόμη και σε ιταλικά επιχειρηματικά έγγραφα μέχρι πρόσφατα ο Musto ανέφερε ότι ήταν κάτοχος διδακτορικού τίτλου από το Ινστιτούτο Τεχνολογίας της Μασαχουσέτης. Σε ορισμένες περιπτώσεις, η βιογραφία του ισχυρίστηκε επίσης ότι κατέχει ένα MBA από το Πανεπιστήμιο της Νέας Υόρκης. Ο Wired ανέφερε ότι σε κανένα από τα πανεπιστήμια δεν υπήρχαν αρχεία που να είναι ο Musto καταγεγραμμένος ενώ αργότερα ο Musto παραδέχτηκε ότι ποτέ του δεν κατείχε αυτούς τους τίτλους.

Το Arduino είναι υλικό ανοικτού κώδικα. Τα σχέδια αναφοράς υλικού διανέμονται υπό την CreativeCommonsAttributionShare-Alike 2.5 άδεια και είναι διαθέσιμα στην ιστοσελίδα του Arduino όπως και κάποια σχέδια και τα αρχεία παραγωγής για κάποιες εκδόσεις του υλικού. Ο πηγαίος κώδικας του IDE διανέμεται υπό την GeneralPublicLicense (**GNU**) v2.Αν και τα σχέδια υλικού και λογισμικού είναι διαθέσιμα υπό την μορφή copyleft αδειών (άδεια πλήρους χρήσης, παραμετροποίησης, διανομής) οι προγραμματιστές ζήτησαν το όνομα του Arduino να αναφέρεται αποκλειστικά και μόνο για το επίσημο προϊόν και να μην χρησιμοποιείται για παράγωγα έργα χωρίς άδεια.Το επίσημο έγγραφο πολιτικής σχετικά με τη χρήση του ονόματος Arduino τονίζει ότι το σχέδιο είναι ανοικτό για να ενσωματώσει την εργασία των άλλων στο επίσημο προϊόν.Αρκετά προϊόντα συμβατά με το Arduino που κυκλοφορούν στο εμπόριο έχουν αποφύγει το όνομα του έργου χρησιμοποιώντας διάφορα ονόματα που λήγουν στο -**duino**.

Shields

Τα περισσότερα Arduino αποτελούνται από μικροελεγκτή Atmel 8-bit AVR (ATmega8 , ATmega168, ATmega328, ATmega1280, ATmega2560) με ποικίλες ποσότητες flash μνήμης, pins και χαρακτηριστικών.Τα Arduino χρησιμοποιούν pins μονής ή διπλής γραμμής ή θηλυκές κεφαλές που διευκολύνουν τις συνδέσεις ,τον προγραμματισμό και την ενσωμάτωση τους σε άλλα κυκλώματα. Η συνδέσεις αυτές μπορούν να γίνου με τα πρόσθετα κυκλώματα που ονομάζονται **Shields**.Τα shields μπορούν να παρέχουν έλεγχο στα motors, GPS, Ethernet, LCD εικόνας ή breadboarding (προτυποποίησης)

Πολλά shields μπορούν να συνδεθούν ακόμη και σε μορφή στοίβας και να επικοινωνούν μέσω του σειριακού διαύλου I²C (**Inter-Integrated Circuit**).Τα περισσότερα Arduino έχουν έναν γραμμικό μετασχηματιστή 5V και ένα κρυστάλλινο ταλαντωτή 16 MHz ή μια κεραμική αντίσταση.

Arduino Boards

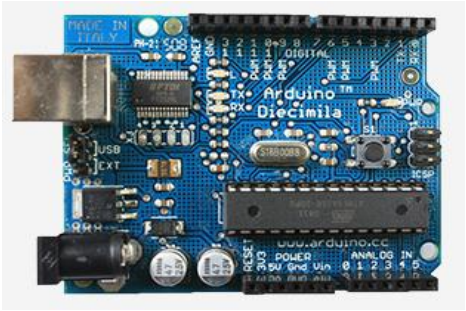
Τα Arduinoέχουνπροεγκατεστημένα από το εργοστάσιο με ένα bootloader ο οποίος έχει ως σκοπό την απλοποίηση της μεταφόρτωσης των προγραμμάτων στην μνήμη flash που βρίσκεται ενσωματωμένη στον μικροελεγκτή. Ο προεπιλεγμένος bootloader του ArduinoUno είναι ο optibootbootloader.Οι τρέχων μικροελεγκτές Arduino προγραμματίζονται μέσω της θύρας USB (UniversalSerialBus) χρησιμοποιώντας προσαρμογείς USB-to-serial όπως το τσιπ FTDIFT232. Κάποια μοντέλα όπως το ArduinoMini και το μοντέλο Boarduino χρησιμοποιούν άλλες μεθόδους όπως αποσπώμενους αντάπτορες USB-to-serial , Bluetooth και άλλες μεθόδους.

Το αρχικό Arduino κατασκευάστηκε από την ιταλική εταιρεία Smart Projects ενώ κάποια επώνυμα προϊόντα Arduino έχουν σχεδιαστεί από τις αμερικανικές εταιρείες SparkFun Electronics και Adafruit Industries. Από το 2016, 17 διαφορετικές εκδόσεις Arduino έχουν βγει στην αγορά.

- **Arduino RS232**

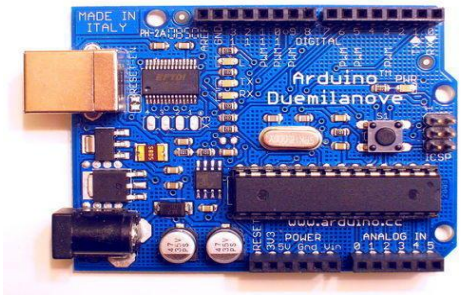
Εικόνα 1.10RS232

- **Arduino Diecimila**



Εικόνα 1.11 Diecimila

- **Arduino Duemilanove**



Εικόνα 1.12 Duemilanove

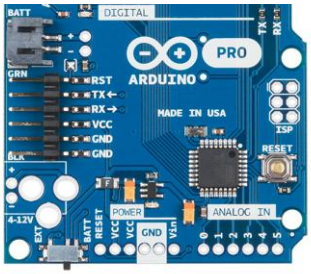
- **Arduino Uno R2**

Εικόνα 1.12 Uno R2

- **Arduino Leonardo**

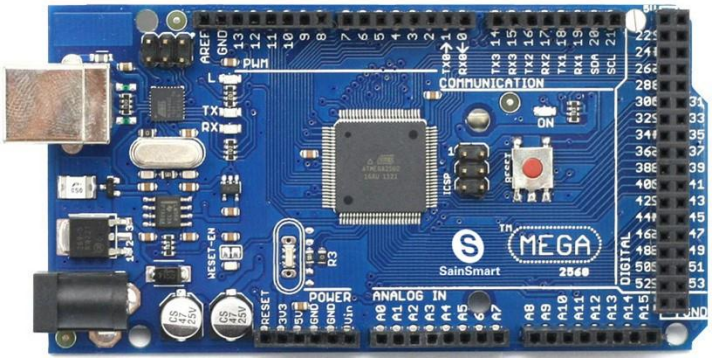
Εικόνα 1.13 Leonardo

- **Arduino Pro**



Εικόνα 1.14 Pro

- **Arduino Mega 2560**

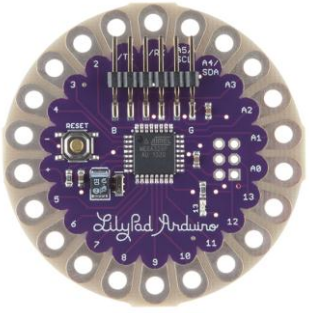


Εικόνα 1.15 Mega 2560

- **Arduino Nano**

Εικόνα 1.16 Nano

- **Arduino LilyPad**

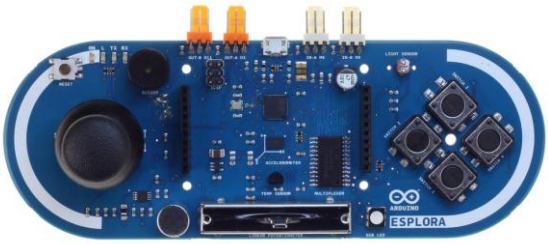


Εικόνα 1.17 LilyPad

- **Arduino Robot**

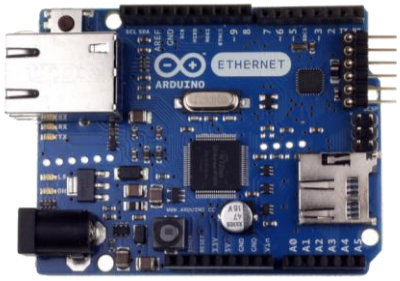
Εικόνα 1.18 Robot

- **Arduino Esplora**



Εικόνα 1.19 Esplora

- **Arduino Ethernet**

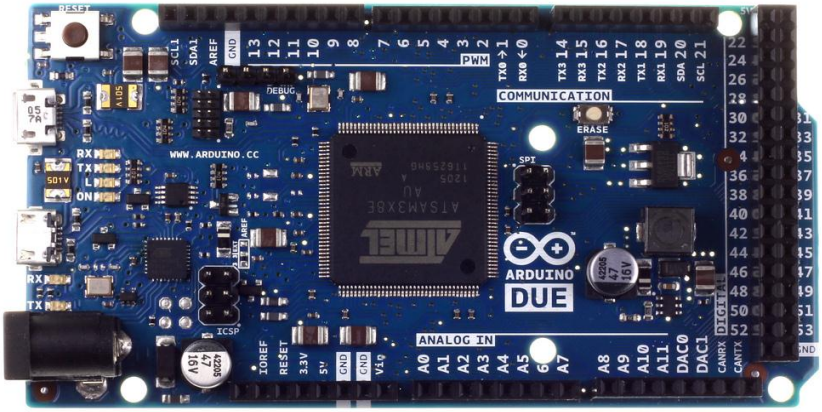


Εικόνα 1.20 Ethernet

- **Arduino Yun**

Εικόνα 1.21 Yun

- **Arduino Due**

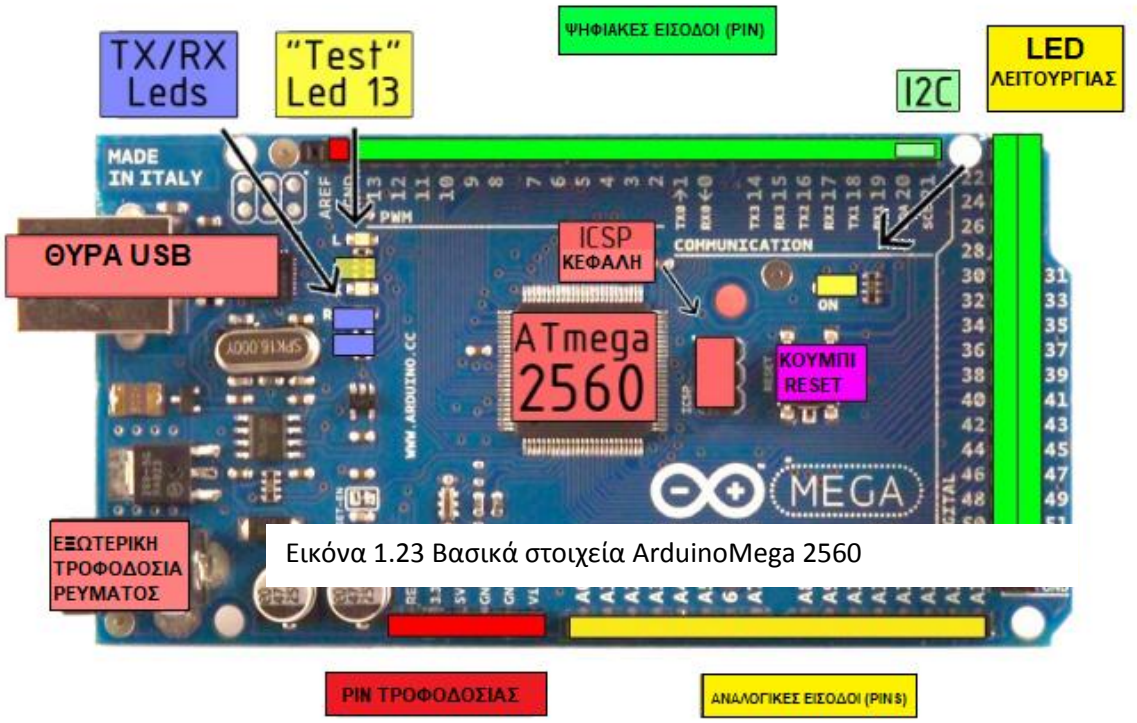


Εικόνα 1.22 Due

ArduinoMega 2560

Η υλοποίηση της εργασίας μας θα γίνει με το ArduinoMega 2560 , για αυτό θα εμβαθύνουμε στα χαρακτηριστικά τον τρόπο λειτουργίας του.

Το Arduino Mega 2560 R3 (έκδοση 3) είναι μια πλακέτα βασισμένη στον μικροελεγκτή ATmega2560. Διαθέτει 54 ψηφιακές εισόδους/εξόδους (15 από αυτές με δυνατότητα PWM), 16 αναλογικές εισόδους, 4 UARTs (σειριακές συνδέσεις με pc ή άλλα Arduino), ρολόι στα 16 MHz,256 KB flash μνήμη, 8 KB SRAM και 4 KB EEPROM καθώς και USB, υποδοχή τροφοδοσίας, υποδοχή ICSP και κουμπί Reset.



Εικόνα 1.23 Βασικά στοιχεία ArduinoMega 2560

ToArduinoMega 2560 περιέχει μια επαναφερόμενη πολυφίσα η οποία προστατεύει τις θύρες usb του υπολογιστή μας από βραχυκυκλώματα και υπερφορτώσεις. Αν και οι περισσότεροι υπολογιστές παρέχουν μια εσωτερική προστασία, η πολυφίσα παρέχει μια επιπλέον ασπίδα προστασίας. Εάν περάσει ρεύμα μεγαλύτερο της τάξης των 500mA στην θύρα usb η φύσα αυτόματα διακόπτει την σύνδεση μέχρι το βραχυκύκλωμα ή η υπερφόρτωση να εξαλειφθεί.

Η τροφοδοσία του Arduino μπορεί να γίνει μέσω την θύρας usb ή μέσω εξωτερική τροφοδοσίας η οποία επιλέγεται αυτόματα. Η εξωτερική τροφοδοσία μπορεί να γίνει είτε από ένα μετατροπέα Ac-to-Dc (εναλλασσόμενο-σε-συνεχές) ρεύμα ή από

μπαταρία. Ο μετατροπέας μπορεί να συνδεθεί συνδέοντας μια 2.1mm με κεντρικό θετικό πόλο φύσα στην θύρα εξωτερικής τροφοδοσίας του Arduino. Οι πόλοι τις μπαταρίας αντίστοιχα μπορούν να συνδεθούν στις κεφαλές (pins) GND(γείωση) και Vin του Arduino.

Ο μικροελεγκτής αυτός μπορεί να λειτουργήσει με εξωτερική τροφοδοσίας 6 έως 20 Volts. Εάν εφαρμοστεί τάση μικρότερη από 7V το pin 5V μπορεί να παρέχει λιγότερα από 5 volt και το κύκλωμα να γίνει ασταθές. Εάν εφαρμοστού πάνω από 12V ο ρυθμιστής τάσης μπορεί να υπερθερμανθεί και να κάνει ζημία στο κύκλωμα. Η συντεινόμενη τάση λειτουργίας είναι από 7 έως 12 Volt.

Τα pin τροφοδοσίας είναι τα εξής:

- **Vin**: Η τάση τροφοδοσίας του κυκλώματος όταν χρησιμοποιείται εξωτερική τροφοδοσία.
- **5V**: Το pin αυτό δίνει 5 volt από τον ρυθμιστή τάσης του Arduino.
- **3V3**: Το pin αυτό δίνει 3.3 volt από τον ρυθμιστή τάσης του Arduino με μέγιστη κατανάλωση ρεύματος 50mA.
- **GND**: Τα pin της γείωσης
- **IOREF**: Το pin αυτό παρέχει την βέλτιστη αναφορά τάσης όπου λειτουργεί ο μικροελεγκτής. Ένα σωστά ρυθμισμένο Shield μπορεί να διαβάσει το IOREFpin και να διαλέξει την βέλτιστη τάση τροφοδοσίας του.

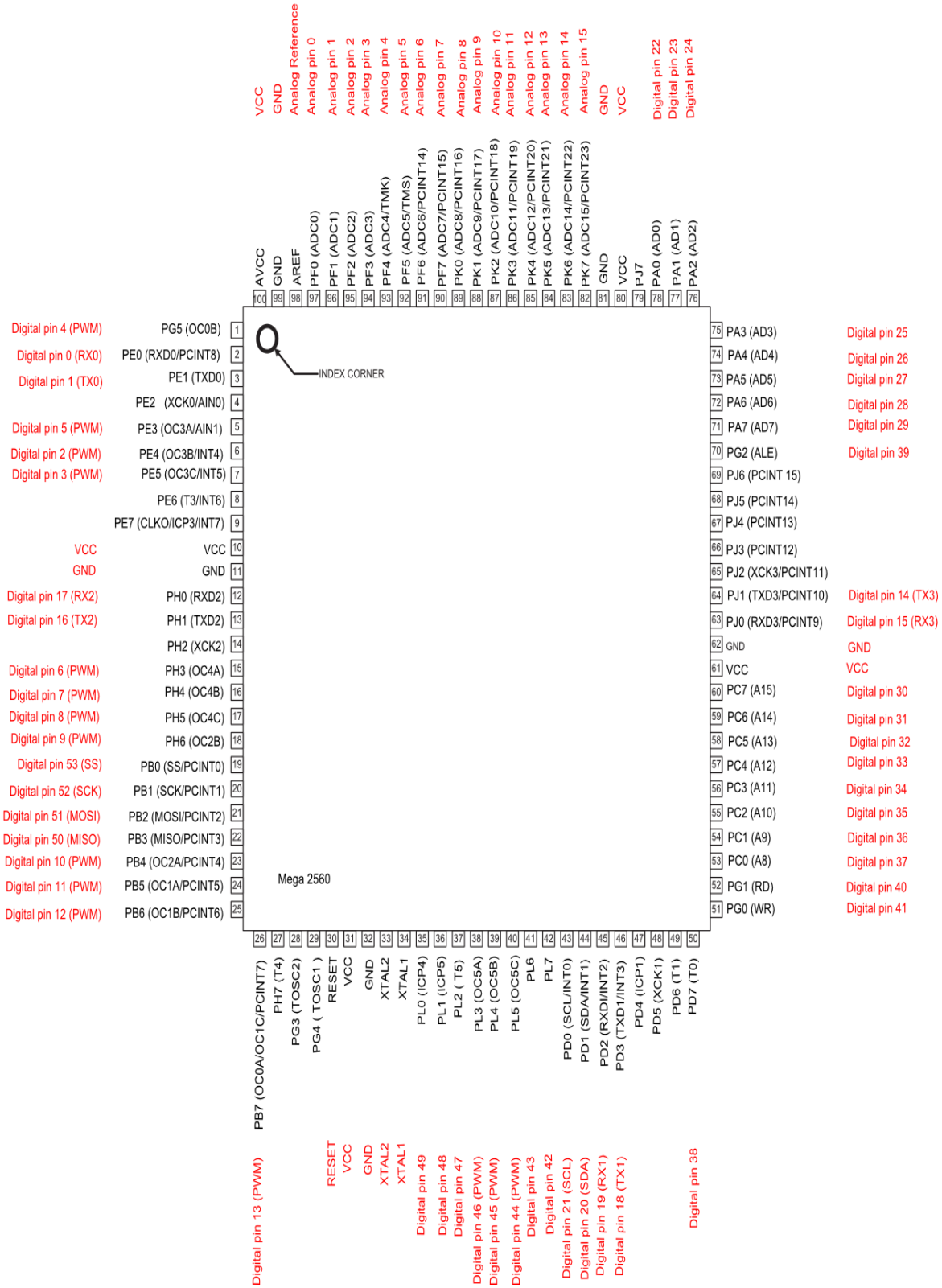
Arduino Mega 2560 Pin Mapping

Κάθε ένα από τα 54 Ψηφιακά pins του ArduinoMega μπορεί να χρησιμοποιηθεί σας είσοδος ή σαν έξοδος χρησιμοποιώντας τις pinMode(),digitalWrite(), and digitalRead() λειτουργίες. Η τάση λειτουργίας του είναι 5 volt και κάθε pin μπορεί να δεχθεί ή να παρέχει 20mA ως συνιστάμενο ρεύμα λειτουργίας. Έχουν εσωτερικές pill-up αντιστάσεις των 20-50 kΩ. Η μέγιστη τιμή των 40mA είναι η μέγιστη τιμή που δεν πρέπει να ξεπεραστεί για την αποφυγή μόνιμης ζημίας στον μικροελεγκτή.

Πάνω στο Arduino υπάρχουν και 16 αναλογικές είσοδοι, κάθε μία από τις οποίες παρέχει 10 bit ανάλυσης (δλδ. 1024 διαφορετικές τιμές). Η εργοστασιακή τάση λειτουργίας του είναι από 0 έως 5 volts, αν και υπάρχει η δυνατότητα αλλαγής της μέγιστης τιμής της τάσης χρησιμοποιώντας το AREFpin και την λειτουργία analogReference().

Άλλα 2 pin που βρίσκουμε στο Arduino είναι:

- **AREF**: Το pin που όπως είπαμε χρησιμοποιείτε με την λειτουργία analogReference() για την αλλαγή της τάσης των αναλογικών pin.
- **Reset**: Το οποίο επαναφέρει τον μικροελεγκτή στην αρχική του κατάσταση.



Εικόνα 1.24 ArduinoMega 2560 Pin Mapping

Κάποια pin έχουν κάποιες ιδιαίτερες λειτουργίες:

- **Serial**: 0 (RX) και 1 (TX); Serial 1: 19 (RX) και 18 (TX); Serial 2: 17 (RX) και 16 (TX); Serial 3: 15 (RX) και 14 (TX). Χρησιμοποιούνται για να λάβουν (RX) και να μεταδώσουν (TX) TTL σειριακά δεδομένα. Τα pins 0 και 1 είναι ενωμένα με τα αντίστοιχα pin του ATmega16U2 USB-to-TTLSerialchip.
- **ExternalInterrupts**: 2 (interrupt 0), 3 (interrupt 1), 18 (interrupt 5), 19 (interrupt 4), 20 (interrupt 3), and 21 (interrupt 2). Αυτά ταpin μπορούν να ρυθμιστούν να δίνουν ερεθίσματα σε μια παρέμβαση (interrupt)σε ένα χαμηλό επίπεδο ή να αλλάζουν σε επίπεδο.
- **PWM**: 2 σε 13 και 44 σε 46. Παρέχουν 8-bitPWMέξοδο με την βοήθεια τηςanalogWrite() λειτουργίας.
- **SPI**: 50 (MISO), 51 (MOSI), 52 (SCK), 53 (SS). Αυτά τα pin υποστηρίζουνSPIεπικοινωνία χρησιμοποιώντας την SPIβιβλιοθήκη . ΤαSPIpinsείναι διαχωρισμένα έξω στην ICSPκεφαλή, η οποία είναι συμβατή με το Arduino /GenuinoUnoκαι τα παλαιότεραDuemilanovekaιDiecimilaArduino.
- **LED**: 13. Είναι ένα ενσωματωμένοLEDσυνδεμένο με το ψηφιακόpin 13. Όταν η τιμή του pinείναιHIGH,τοLEDείναι αναμμένο και όταν είναι σε LOW, είναι σβηστό.
- **TWI**: 20 (SDA) και 21 (SCL). ΥποστηρίζουνTWIεπικοινωνία χρησιμοποιώντας την Wireβιβλιοθήκη.

Το ArduinoMega 2560 χρησιμοποιείδιάφορες λειτουργίες για την επικοινωνία με τον υπολογιστή, άλλα Arduino και μικροελεγκτές. Παρέχει 4 κυκλώματα UART για TTL(5V) σειριακή επικοινωνία. Το ArduinoIDE παρέχει ένα serialmonitor το οποίο επιτρέπει την αποστολή από και προς το Arduino μας. Τα LEDRX και TX του κυκλώματος θα αναβοσβήνουν καθώς μεταδίδονται δεδομένα από το ATmega8U2/ATmega16U2 chip και την USB σύνδεση με τον υπολογιστή.

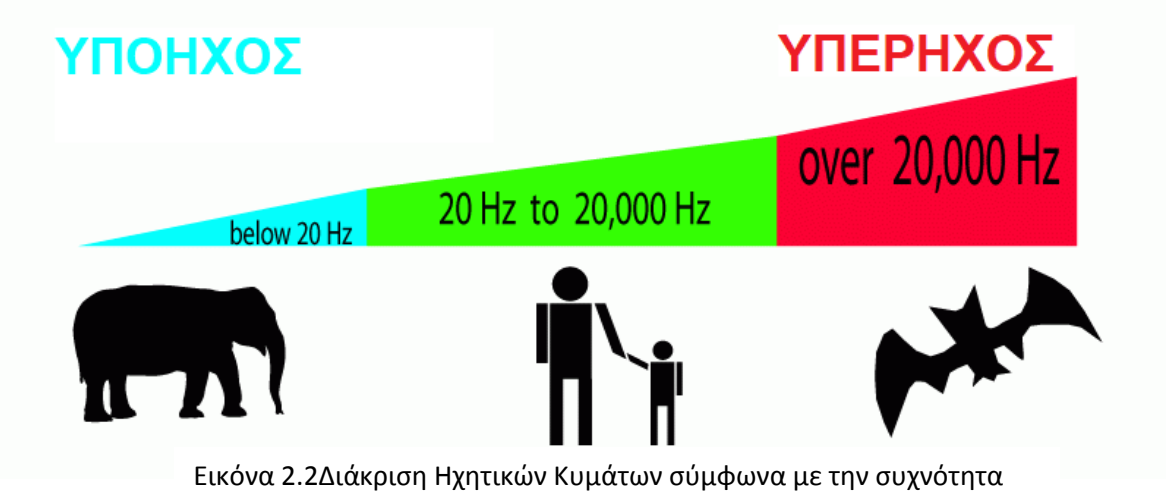
Η SoftwareSerial βιβλιοθήκη επιτρέπει την ψηφιακή επικοινωνία όλων των ψηφιακών pins του ArduinoMega 2560. Ακόμη το Arduino μας υποστηρίζει TWI και SPI επικοινωνία. Το Arduino (IDE) περιέχει τη Wire βιβλιοθήκη για την απλοποίηση της χρήσης του TWI διαύλου. Για SPI επικοινωνία υπάρχει η βιβλιοθήκη SPI.

2. Αισθητήρας HC-SR04

Εικόνα 2.1Αισθητήρας HC-SR04

Ορισμός του υπέρηχου

Υπέρηχος (ultrasound) ονομάζεται το μηχανικό κύμα με συχνότητα μεγαλύτερη από αυτήν που μπορεί να ακούσει ο άνθρωπος (περίπου 20.000 Hz). Με άλλα λόγια ο υπέρηχος είναι ένας ήχος τόσο ψηλός που δεν μπορούμε να τον ακούσουμε, καθώς το ανθρώπινο αυτί έχει κάποια όρια και δεν μπορεί να ακούσει πολύ ψηλούς ή χαμηλούς ήχους. Ωστόσο, άλλα ζώα είναι ικανά να ακούσουν υπερήχους, όπως ο σκύλος, ενώ οι νυχτερίδες χρησιμοποιούν για να "βλέπουν" την νύχτα.Η φυσική των υπερήχων είναι ίδια με των ηχητικών κυμάτων.



Εικόνα 2.2Διάκριση Ηχητικών Κυμάτων σύμφωνα με την συχνότητα

Αρχή μέτρησης απόστασης με υπέρηχους

Ο εκπομπός υπερήχων εκπέμπει ένα υπερηχητικό κύμα προς μία κατεύθυνση και η μέτρηση του χρόνου ξεκινάει από την στιγμή που θα ξεκινήσει. Ο υπέρηχος απλώνεται στο αέρα και επιστρέψει άμεσα μόλις βρει εμπόδιο στον δρόμο του. Ολοκληρώνοντας ο δέκτης υπέρηχων σταματάει να μετράει μόλις επιστραφεί το ανακλώμενο κύμα. Η ταχύτητα διάδοσης του υπέρηχου είναι 340 m/s στον αέρα και 1500 m/s στο νερό.

Η αρχή μέτρησης της απόστασης με υπέρηχους χρησιμοποιεί την ήδη γνωστή ταχύτητα διάδοσης στις συνθήκες που υπάρχουν , μετρώντας τον χρόνο από την στιγμή που ξεκίνησε η διάδοση του υπερηχητικού κύματος μέχρι την στιγμή που βρήκε εμπόδιο στην πορεία του και μετά υπολογίζοντας την απόσταση μεταξύ του εκπομπού και του εμποδίου σύμφωνα με τον χρόνο και την ταχύτητα διάδοσης. Η Αρχή μέτρησης απόστασης με υπέρηχους είναι η ίδια με τα συστήματα Radar.

Η χρήση της τεχνολογίας των υπερήχων είναι κάτι το οποίο αναπτύχθηκε τις τελευταίες δεκαετίες. Με τα τεράστια πλεονεκτήματα των υπέρηχων η ανάπτυξη των ηλεκτρικών τεχνολογιών και ειδικότερα καθώς η τεχνολογία των ημιαγωγών υψηλής ισχύος ακμάζει η χρήση των υπέρηχων αυξάνεται ευρέως στους εξής τομείς:

- Υπερηχητική μέτρηση απόστασης, βάθους και πάχους
- Υπερηχητικές δοκιμές
- Υπερηχητικές απεικονίσεις
- Υπερηχητική μηχανουργία όπως γυάλισμα και τόρνευση
- Υπερηχητικοί καθαρισμοί
- Υπερηχητικές συγκολλήσεις

Χαρακτηριστικά HC-SR04:

- *Σταθερή απόδοση*
- *Ακριβής μέτρηση απόστασης*
- *Υψηλή πυκνότητα μετρήσεων*
- *Μικρό σφάλμα*

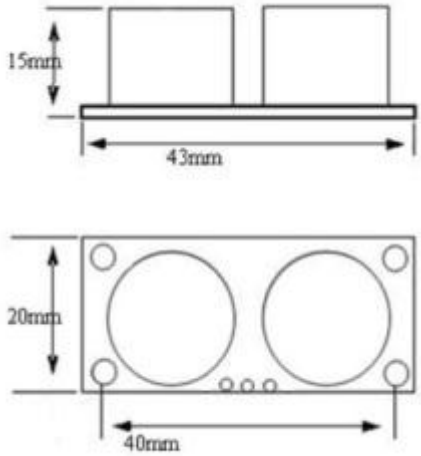
Τομείς χρήσης:

- Ρομποτικά περιθώρια
- Μέτρηση απόστασης αντικειμένων
- Εντοπισμός επιπέδου
- Δημόσια ασφάλεια
- Αισθητήρες παρκαρίσματος

PinoutHC-SR04

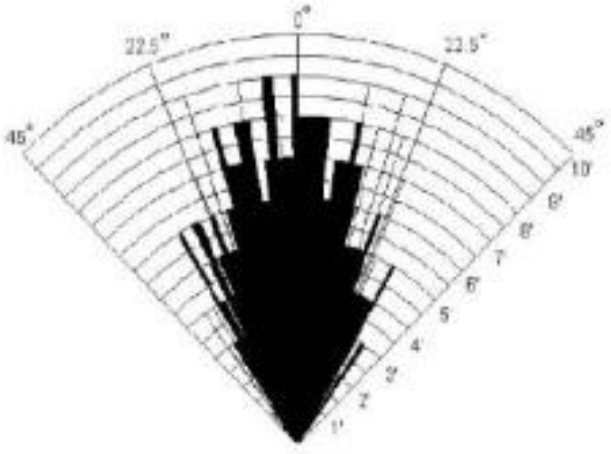
Σύμβολο Pin	Λειτουργία
VCC	+5VDC
Trig	Trigger input of Sensor (Είσοδος εκπομπού αισθητήρα)
Echo	Echo output of Sensor (Έξοδος δέκτη αισθητήρα)
GND	Γείωση

Οι διαστάσεις του αισθητήρα είναι 43mm μήκος, 20mm πλάτος και 15mm ύψος



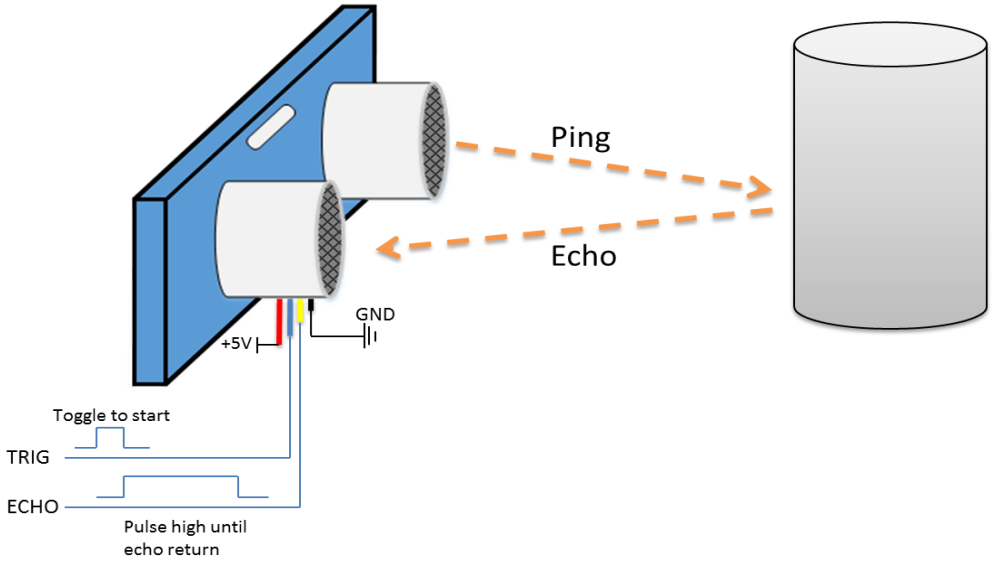
Εικόνα 2.3 Φυσικές διαστάσεις HC-SR04

Και αποδίδει καλύτερα στις 15^{o} μοίρες



Εικόνα 2.4 Μοίρες απόδοσης ορθής λειτουργίας HC-SR04

Ηλεκτρικές παράμετροι	HC-SR04
Τάση λειτουργίας	DC-5V
Ρεύμα λειτουργίας	15mA
Συχνότητα λειτουργίας	40KHZ
Μέγιστη απόσταση κάλυψης	4 m (μέτρα)
Ελάχιστη απόσταση κάλυψης	2 cm (εκατοστά)
Γωνία μέτρησης	15 Degree (μοίρες)
Σήμα εισόδου εκπομπού	10 us TTL pulse (παλμού)
Σήμα εξόδου δέκτη	Έξοδος TTL επιπέδου σήμα, ανάλογα με την εμβέλεια
Διαστάσεις	43*20*15mm



Εικόνα2.5 HC-SR04 Ping – Echo operation

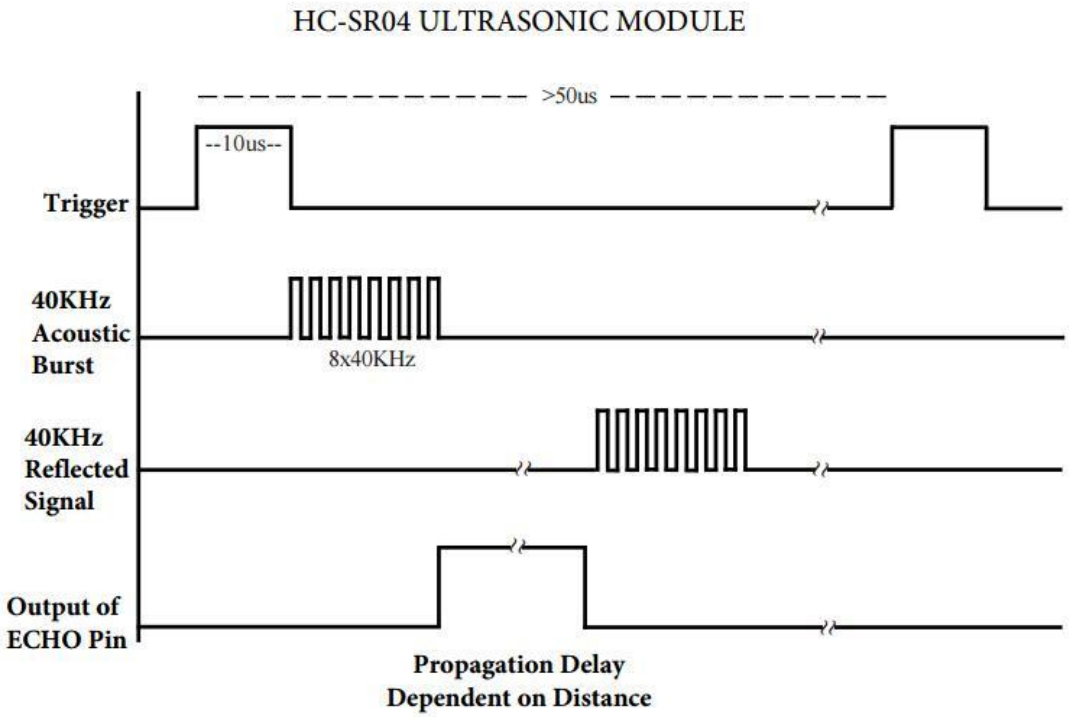
Παράμετροι	Ελάχιστη	Τυπική	Μέγιστη	Μονάδα
Τάση Λειτουργίας	4.50	5.0	5.5	V
Ρεύμα ηρεμίας	1.5	2	2.5	mA
Ρεύμα λειτουργίας	10	15	20	mA
Υπερηχητική Συχνότητα	-	40	-	kHz

Τρόπος λειτουργίας

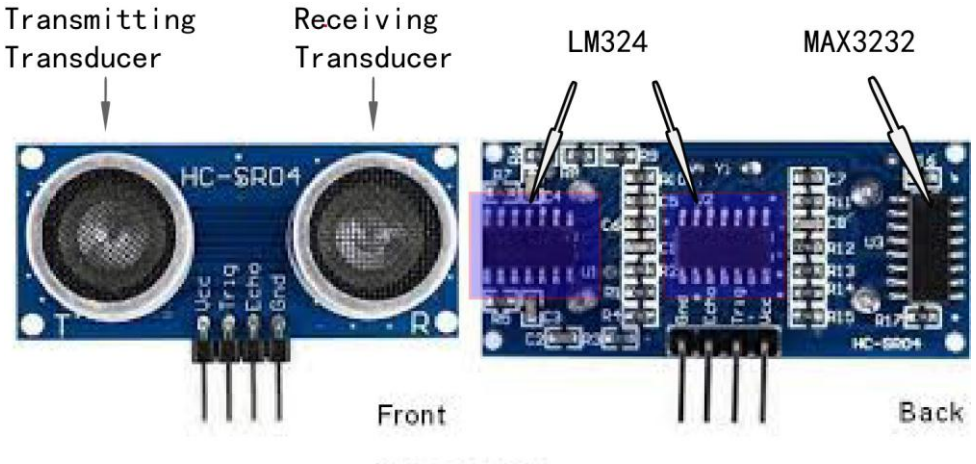
Για να ξεκινήσει η μέτρηση, topin**Trig** του SR04 πρέπει να λάβει έναν παλμό υψηλού (**+5V**) για τουλάχιστον 10 μS, με αυτό θα ξεκινήσει ο αισθητήρας την λειτουργία τουμεταδίδοντας 8 κύκλους τετραγωνικού παλμούτων 40kHz και θα περιμένει την ανακλώμενη τετραγωνικό παλμό. Όταν ο αισθητήρας ανιχνεύσει υπερήχους από το δέκτη, θα θέσει το pin **Echo** σε υψηλό **(+5V**) και θα καθυστερήσει για μια περίοδο (πλάτος) σε σχέση με την απόσταση. Για να λάβετε την απόσταση, μετρήστε το πλάτος (Ton) του pin Echo.

Χρόνος = Πλάτος παλμού Echo, σε μS

- Απόσταση σε εκατοστά = Χρόνος / 58
- Απόσταση σε ίντσες = Χρόνος / 148
- Ή μπορούμε να χρησιμοποιήσετε την ταχύτητα του ήχου, η οποία είναι 340m / s στον αέρα ή 1500 m / s στο νερό.



Εικόνα2.6HC-SR04 Λειτουργία σε διάστημα >50us



Εικόνα 2.7 Στοιχεία αισθητήρα HC-SR04

Στο μπροστινό μέρος του ανιχνευτή υπερήχων υπάρχουν δύο μεταλλικοί κύλινδροι. Αυτοί είναι μετατροπείς. Οι μετατροπείς, μετατρέπουν τις μηχανικές δυνάμεις σε ηλεκτρικά σήματα. Στον ανιχνευτή υπερήχων, υπάρχει ένας μετατροπέας μετάδοσης και ένας μετατροπέας λήψης. Ο μετατροπέας μετάδοσης μετατρέπει ένα ηλεκτρικό σήμα στον υπερηχητικό παλμό και ο μετατροπέας λήψης μετατρέπει τον ανακλώμενο υπερηχητικό παλμό πίσω σε ένα ηλεκτρικό σήμα. Αν κοιτάξετε στο πίσω μέρος του HC-SR04, θα δείτε ένα ολοκληρωμένο κύκλωμα πίσω από τον αισθητήρα μετάδοσης που ονομάζεται MAX3232. Αυτό είναι το ολοκληρωμένο κύκλωμα που ελέγχει τον μετατροπέα μετάδοσης. Πίσω από τον μετατροπέα λήψης είναι ένα ολοκληρωμένο κύκλωμα που ονομάζεται LM324. Πρόκειται για ένα Quad Op-Amp (ενισχυτής) που ενισχύει το σήμα που παράγεται από τον μετατροπέα λήψης σε ένα σήμα που είναι αρκετά ισχυρό για να μεταδοθεί στο Arduino.

3. Σερβοκινητήρας

Στον τομέα του ελέγχου κίνησης και ειδικότερα σε εφαρμογές ελέγχου θέσεως ταχύτητας και ροπής άξονα χρησιμοποιούνται εδώ και αρκετά χρόνια ειδικοί κινητήρες που λέγονται σερβοκινητήρες. Η επιτυχία στην δημιουργία των σερβοκινητήρων έγκειται στην δυνατότητα εκτός από την απλή ρύθμιση των στροφών και της ταχύτητας που γίνονται και στους συνήθεις κινητήρες με ρυθμιστές στροφών και inverters να γίνεται πραγματικός έλεγχος της θέσης του άξονα του κινητήρα (positioning). Η παραπάνω δυνατότητα γίνεται εφικτή μέσω της προσθήκης ενός συγκεκριμένου αισθητηρίου στο σώμα του κινητήρα (resolver ή encoder) που ελέγχει με συγκεκριμένες διαδικασίες την ταχύτητα και τη θέση του άξονα του κινητήρα. Με την εξέλιξη αυτή η τεχνολογία πέρασε από τον κλασσικό έλεγχο της ταχύτητας (speed control) στο συνολικό έλεγχο της κίνησης (motion control). Αυτή η εφαρμογή αποτέλεσε κομβικό σημείο στην εξέλιξη της ρομποτικής.

Οι σερβοκινητήρες δεν μπορούν να επιτελέσουν την ουσιώδη λειτουργία τους εκτελώντας μεμονωμένες κινήσεις, αλλά αποτελούν το βασικό στοιχείο ενός συνολικού συστήματος αυτοματισμού που επιτελεί motion control.

Εικόνα 3.1 FEETECH Servo Motors Series

Στο συνολικό σερβοσύστημα εντάσσονται εκτός από το σερβοκινητήρα, η μονάδα ελέγχου που στην πλειοψηφία των περιπτώσεων είναι ένα PLC και ο servodriveπου αποτελεί τον ενδιάμεσο κρίκο μεταξύ της μονάδας ελέγχου και του σερβοκινητήρα. Οι σερβοκινητήρες είναι οι κινητήρες που χρησιμοποιούνται στα Συστήματα Αυτομάτου Ελέγχου, χωρίς φυσικά να ταυτίζονται με τους κοινούς κινητήρες παρόλο που μοιάζουν κατασκευαστικά. Οι σερβοκινητήρες διαφέρουν από τους άλλους κινητήρες στο ότι ενσωματώνουν ένα σύστημα ανάδρασης το οποίο χρησιμοποιείται σε συνδυασμό με έναν σερβομηχανισμό οδήγησης με σκοπό να ελεγχθεί είτε η ροπή είτε η ταχύτητα.

Ο σερβομηχανισμός οδήγησης συνίσταται στον σερβοενισχυτή (servodrive) που αποτελεί ένα ξεχωριστό στοιχείο του σερβοσυστήματος. Ωστόσο, το σύστημα ανάδρασης είναι ενσωματωμένο στο σερβοκινητήρα. Οι περισσότεροι σερβοκινητήρες διαθέτουν μια ενσωματωμένη παλμογεννήτρια (encoder ή pulse generator), αυξητικού ή απόλυτου τύπου (incremental ή absolute type). Αν η παλμογεννήτρια είναι αυξητικού τύπου τότε παράγουν ένα συγκεκριμένο αριθμό παλμών ανά περιστροφή του άξονα του σερβοκινητήρα, ενώ αν είναι απολύτου θέσεως τότε δίνουν στην έξοδό τους συγκεκριμένο αριθμό ως αποτέλεσμα μιας ακολουθίας bit. Και στις δύο περιπτώσεις τα σήματα μπορούν να μεταφερθούν στη μονάδα ελέγχου, όπου με τον κατάλληλο προγραμματισμό να επιτευχθεί ο βέλτιστος έλεγχος του σερβοκινητήρα. Πιο συγκεκριμένα, ο έλεγχος αυτός μπορεί να αφορά είτε στη θέση περιστροφής του άξονα ως προς τις 360 μοίρες που συνιστούν μια πλήρη περιστροφή, είτε στην ταχύτητα περιστροφής του άξονα, είτε στην αναπτυσσόμενη σε αυτόν ροπή. Αν εξετάσει κανείς τους σερβοκινητήρες από τη σκοπιά των ηλεκτροκινητήρων, θα διαπιστώσει ότι, από λειτουργική άποψη κύριο γνώρισμα τους είναι η ικανότητά τους να αναπτύσσουν μεγάλες επιταχύνσεις στην περίπτωση της πλήρους ακινησίας, όταν δηλαδή παρουσιάζεται μικρή ροπή αδράνειας και μεγάλη ροπή στρέψης.

Για την επίτευξη των προηγούμενων λειτουργιών πρέπει να τηρούνται οι ακόλουθες προϋποθέσεις:

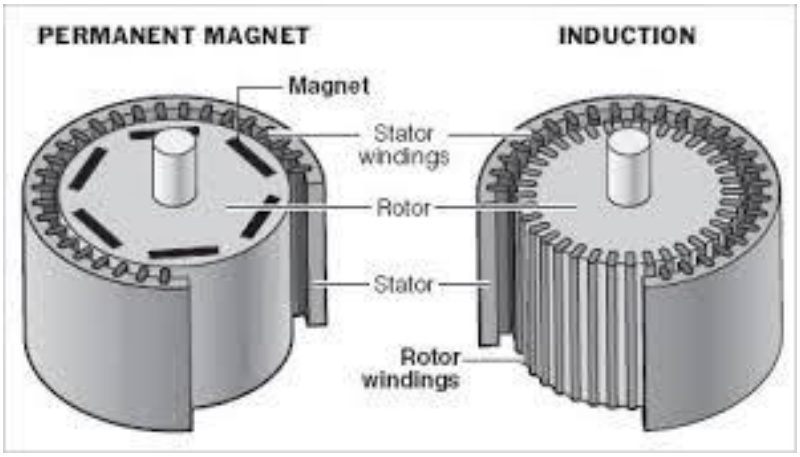
- Ο ρότορας να έχει μεγάλο μήκος και μικρή διάμετρο.
- Να υπάρχουν περιελίξεις αντισταθμίσεως οι οποίες επιτρέποντας ανάπτυξη μεγαλύτερων ρευμάτων αυξάνουν τη ροπή στρέψης.
- Για μικρής ισχύος κινητήρες προβλέπεται μόνιμος μαγνήτης μέσα στους πόλους του οποίου και γύρω από ένα μόνιμο στέλεχος (όπως στα όργανα κινητού πλαισίου) περιστρέφεται το τύλιγμα του ρότορα.
- Να είναι μειωμένη η σταθερά χρόνου L.R του τυλίγματος του ρότορα.

Η επιλογή ενός σερβοκινητήρα γίνεται έχοντας υπόψη ότι η ισχύς του θα πρέπει να καλύπτει την ισχύ του φορτίου (ωφέλιμη) και τις τριβές (απώλειες) της διάταξης. Πέραν αυτού ο σερβοκινητήρας πρέπει να λειτουργεί στις επιθυμητές ταχύτητες και συνάμα να παρέχει την απαραίτητη επιτάχυνση στο ρότορα και στο φορτίο. Οι σερβοκινητήρες διακρίνονται σε ηλεκτρικούς AC και DC, σε πνευματικούς και υδραυλικούς.

Οι μονοφασικοί σερβοκινητήρες συνεχούς ρεύματος διακρίνονται στους εξής τέσσερις τύπους:

- Ο πρώτος είναι αυτός που τα τυλίγματα του στάτορα τροφοδοτούνται από πηγή σταθερής τάσεως ή ρεύματος, ενώ το τύλιγμα του ρότορα από μια τάση ελέγχου. Οι σερβοκινητήρες αυτοί είναι γνωστοί σαν ελεγχόμενοι από το ρότορα. Σε αυτούς τους σερβοκινητήρες αν διατηρούμε σταθερή την τάση ελέγχου Vε η ροπή στρέψης μικραίνει γραμμικά σε συνάρτηση με την αύξηση της γωνιακής ταχύτητας ω του κινητήρα.
- Ο δεύτερος τύπος σερβοκινητήρα είναι ο ελεγχόμενος από το στάτορα. Σ' αυτόν τον τύπο το τύλιγμα του ρότορα τροφοδοτείται από μια πηγή σταθερής τάσεως ή ρεύματος ενώ το τύλιγμα του στάτορα από μια τάση ελέγχου. Σε αυτούς τους σερβοκινητήρες η ροπή στρέψης είναι ανεξάρτητη από τη γωνιακή ταχύτητα του στάτορα και εξαρτάται μόνο από τη σταθερά Κ και το ρεύμα του στάτορα. Ωστόσο αν το μαγνητικό υλικό εργάζεται στον κόρο η ροπή στρέψης επηρεάζεται και από τη γωνιακή ταχύτητα του στάτορα και μάλιστα σε πολύ μεγάλες γωνιακές ταχύτητες η ροπή μικραίνει γιατί αυξάνει πάρα πολύ η αν ηλεκτρεγερτική δύναμη.
- Ο τρίτος τύπος είναι ο σερβοκινητήρας με τα τυλίγματα στάτορα και ρότορα σε σύνδεση σειράς: Οι σερβοκινητήρες αυτοί έχουν διπλό τύλιγμα στο στάτορα έτσι που το καθένα να συνδέεται σε σειρά με το τύλιγμα του ρότορα με τη βοήθεια ηλεκτρονόμων. Η ροπή στρέψης του κινητήρα μεταβάλλεται εκθετικά και εξαρτάται από τα μεγέθη του ρεύματος ελέγχου και της γωνιακής ταχύτητας. Είναι πολύ μεγάλη κατά την εκκίνηση οπότε η γωνιακή ταχύτητα είναι μικρή, ενώ μικραίνει απότομα όταν η γωνιακή ταχύτητα μεγαλώνει, χρησιμοποιείται κυρίως εκεί όπου απαιτείται μεγάλη ροπή κατά την εκκίνηση (όπου έχουμε περιστροφή μαζών) αφού η γραμμικότητα δεν παίζει κανένα ρόλο.

- Ένας ιδιαίτερα σημαντικός τύπος σερβοκινητήρας είναι αυτός με μόνιμο μαγνήτη. Ο σερβοκινητήρας του τύπου αυτού έχει αντί για τυλίγματα στάτορα, μόνιμο μαγνήτη, ενώ ο ρότορας έχει κανονικό τύλιγμα μέσα από το οποίο ελέγχεται ο κινητήρας. Ο κινητήρας αυτός μοιάζει πολύ με τους ασύγχρονους κινητήρες παράλληλης διέγερσης και λόγω του μικρού όγκου του χρησιμοποιείται σε Συστήματα Αυτομάτου Ελέγχου πάνω σε αεροπλάνα. Τέλος, ο μικρός όγκος του κινητήρα επιτυγχάνεται μέσω ειδικού κράματος μόνιμου μαγνήτη.



Εικόνα 3.2 Εσωτερικό Servo Motor

Οι μαγνήτες που χρησιμοποιούνται στους σερβοκινητήρες κατηγοριοποιούνται στους παρακάτω βασικούς τύπους:

- Οι κεραμικοί μαγνήτες οι οποίοι αποτελούνται από οξείδιο του σιδήρου και καρβίδιο του βαρίου ή του στροντίου. Οι κεραμικοί μαγνήτες χρησιμοποιούνται σε κινητήρες μικρής σχετικά ισχύος για μη ενεργοβόρες διαδικασίες ελέγχου.
- Οι μαγνήτες AlNiCo, δηλαδή οι μαγνήτες αλουμινίου, νικελίου, κοβαλτίου, οι οποίοι είναι δυνατόν να περιέχουν ίχνη από σίδηρο, χαλκό και τιτάνιο. Σήμερα δε περιλαμβάνονται στο σχεδιασμό των καινούργιων κινητήρων εξαιτίας του υψηλού κόστους και της σχετικά εύκολης απομαγνήτισης τους σε συνθήκες ανοικτού κυκλώματος.
- Οι μαγνήτες Σαμαρίου Κοβαλτίου, οι οποίοι λόγω του υψηλού κόστους χρησιμοποιούνται μόνο σε εφαρμογές στις οποίες η υψηλή θερμοκρασία και η αντοχή σε διάβρωση αποτελούν κρίσιμες παραμέτρους.
- Οι μαγνήτες Νεοδύμιου Σιδήρου Βορίου(NdFeB) είναι οι πιο σύγχρονη γενιά μαγνητών. Οι εξαιρετικές μαγνητικές τους ιδιότητες τους καθιστούν παράλληλα κατάλληλους και για συμπαγείς κατασκευές σε χρήσεις που απαιτούν μικρότερα κόστη κατασκευής. Σημαντικό μειονέκτημα τους είναι ότι διαβρώνονται πολύ εύκολα.

Οι σερβοκινητήρες εναλλασσόμενου ρεύματος μπορούν να είναι διφασικοί ή τριφασικοί. Οι διφασικοί αποτελούνται από δύο τυλίγματα στο στάτη με τέτοια τοποθέτηση, ώστε να παρουσιάζουν διαφορά φάσεως 90 μοιρών και το ρότορα. Το ένα τύλιγμα ονομάζεται τύλιγμα αναφοράς και τροφοδοτείται από μια εναλλασσόμενη τάση σταθερής τιμής, ενώ το άλλο τύλιγμα είναι τύλιγμα ελέγχου και τροφοδοτείται από την τάση ελέγχου. Όταν λοιπόν εφαρμοστούν αυτές οι τάσεις στα τυλίγματα τότε δημιουργείται στρεφόμενο μαγνητικό πεδίο από τα δύο ρεύματα που διαρρέουν τα τυλίγματα και ο ρότορας περιστρέφεται. Ο ρότορας είναι φτιαγμένος από χάλκινες ράβδους που βραχυκυκλώνουν μεταξύ τους (βραχυκυκλωμένος δρομέας). Η ταχύτητα και η διεύθυνση περιστροφής καθορίζονται από το πλάτος και τη φάση της τάσης ελέγχου. Οι κινητήρες εναλλασσόμενου ρεύματος παρουσιάζουν μεγάλη ροπή στρέψης για μικρές γωνιακές ταχύτητες.

Επιπρόσθετα, η σχέση μεταξύ ροπής και γωνιακής ταχύτητας είναι όμοια με αυτήν των σερβοκινητήρων συνεχούς ρεύματος που ελέγχονται από το ρότορα, δηλαδή η ροπή στρέψης μικραίνει γραμμικά σε συνάρτηση με την αύξηση της γωνιακής ταχύτητας. Η λειτουργία ανάδρασης σε έναν σερβοκινητήρα αποσκοπεί στο συνεχή έλεγχο των εντολών θέσης και ταχύτητας που δίνονται προς τον κινητήρα. Αυτό επιτυγχάνεται από τον ενισχυτή του σερβοσυστήματος που αποτελεί και το σύστημα οδήγησης του σερβοκινητήρα (servodrive). Ο «σερβοενισχυτής» αποτελεί τον ενδιάμεσο σταθμό μεταξύ μονάδας ελέγχου και σερβοκινητήρα. Στην πραγματικότητα το servodrive είναι ένα ειδικού τύπου inverter, το οποίο όμως χρησιμοποιείται αποκλειστικά για έλεγχο σερβοκινητήρων και γι’ αυτό έχει σχεδιαστεί έτσι ώστε να επιτελεί μια πολύ ειδική λειτουργία. Η λειτουργία των ενισχυτών του σερβοκινητήρα (servodrives) πιο συγκεκριμένα, αποσκοπεί στο να διατηρεί σταθερές τις απαιτούμενες στροφές, να διατηρεί σταθερή τη ροπή σε όλη την περιοχή στροφών του κινητήρα, αλλά ταυτόχρονα να δίνει τη δυνατότητα της βηματικής κίνησης με απόλυτο έλεγχο των δύο προηγούμενων παραμέτρων.

Αναλυτικότερα, τα συστήματα οδήγησης σερβοκινητήρων (servodrives) επιτελούν τρεις βασικές λειτουργίες:

- Τον έλεγχο της ανάδρασης του «σερβοσυστήματος».
- Τον έλεγχο του κινητήρα.
- Τη μετατροπή ισχύος.

Ο έλεγχος ενός «σερβοσυστήματος» συνίσταται στη ρύθμιση της ταχύτητας και της θέσης ενός κινητήρα. Η ρύθμιση αυτή βασίζεται σε ένα σήμα ανάδρασης. Το βασικότερο «σερβοκύκλωμα» είναι το κύκλωμα της ταχύτητας, του οποίου ο ρόλος είναι η παραγωγή μιας εντολοδότησης για τη ροπή με σκοπό την ελαχιστοποίηση του σφάλματος μεταξύ της αρχικής εντολοδότησης για την ταχύτητα και της ταχύτητας που λαμβάνεται από το σήμα ανάδρασης. Λόγω ότι στους σερβοκινητήρες κατά κανόνα απαιτείται να υπάρχει και έλεγχος θέσης συνήθως προστίθεται ένα κύκλωμα ελέγχου θέσης σε σειρά με το κύκλωμα ελέγχου ταχύτητας.

Στα κυκλώματα ελέγχου ενός σερβοσυστήματος σημαντική παράμετρος είναι και η ρύθμιση της έντασης του σήματος ανάδρασης. Αν αυτή η ρύθμιση γίνει σε ψηλή στάθμη τότε αφενός μεν τα αποτελέσματα είναι καλύτερα, αφετέρου το συνολικό σερβοσύστημα καθίσταται πιο ασταθές. Για το λόγο αυτό χρησιμοποιούνται φίλτρα που συνδέονται σε σειρά με τα κυκλώματα ελέγχου ταχύτητας. Ο έλεγχος του κινητήρα συνίσταται στην παραγωγή μιας ροπής κινητήρα που να ανταποκρίνεται στην εντολοδότηση ροπής που στέλνεται από το κύκλωμα ελέγχου του σερβοσυστήματος. Στους κινητήρες που φέρουν ψήκτρες (κατά κύριο λόγο είναι οι κινητήρες συνεχούς ρεύματος) ο έλεγχος του κινητήρα συνίσταται απλά στον έλεγχο του ρεύματος των τυλιγμάτων καθώς η ροπή του κινητήρα είναι περίπου ανάλογη του ρεύματος των τυλιγμάτων. Τα κυκλώματα ελέγχου ρεύματος είναι όμοια με αυτά του ελέγχου ταχύτητας με τη μόνη διαφορά ότι ενεργούν σε ψηλότερες συχνότητες. Ένα κύκλωμα ελέγχου ρεύματος λαμβάνει την εντολοδότηση ρεύματος και τη συγκρίνει με την τιμή του ρεύματος που λαμβάνεται από το σήμα ανάδρασης παράγοντας μια έξοδο, δηλαδή ένα σήμα ρύθμισης τάσης. Αν ο κινητήρας χρειάζεται να δουλέψει με μεγαλύτερη ροπή τότε αυξάνεται η εφαρμοζόμενη σ' αυτόν τάση μέχρι να επιτευχθεί το επιθυμητό ρεύμα τυλιγμάτων. Τέλος, αναφορικά με τη μετατροπή της ταχύτητας υπάρχουν αλγόριθμοι που στηρίζονται στην ικανότητα της πηγής ισχύος να παράγει το ρεύμα εκείνο που θα ικανοποιεί τις απαιτήσεις που προκύπτουν από τα κυκλώματα ελέγχου ταχύτητας και θέσης.

Εικόνα 3.3 FEETECHFS90Rσυνδεμένο σε τροχό

Η σύνδεση του σερβοενισχυτή με το συνολικό σερβοσύστημα γίνεται συνήθως με μια σειρά θυρών επικοινωνίας που διαθέτει ο ειδικός αυτός inverter. Μια θύρα μπορεί να χρησιμοποιείται για τη σύνδεση του servodrive με την παλμογεννήτρια του σερβοσυστήματος που συνιστά το τοπικό αισθητήριο του στο ελεγχόμενο σημείο της παραγωγικής ή κατασκευαστικής διαδικασίας. Μια άλλη θύρα μπορεί να συνδέεται με Η/Υ, μέσω του οποίου μπορεί να προγραμματίζεται ο σερβοενισχυτής, έτσι ώστε με τον καθορισμό συγκεκριμένων παραμέτρων να επιτυγχάνεται ο βέλτιστος έλεγχος του κινητήρα. Σε περίπτωση που το ο κεντρικός έλεγχος του σερβοσυστήματος υποστηρίζεται από PLC ή DCS, τότε ο σερβοενισχυτής μπορεί να διαθέτει κι άλλες θύρες για την εισαγωγή σημάτων από το PLC ή το DCS. Υπάρχει και η περίπτωση οι σχεδιαστικές επιλογές να απαιτούν τη μεταφορά τοπικών σημάτων μέσω άλλων αισθητηρίων, τερματικών ή ακόμα και μπουτόν στο servodrive χωρίς αυτά προηγουμένως να περνάνε από το PLC ή

το DCS. Και σε αυτήν την περίπτωση χρειάζεται ο σερβοενισχυτής να είναι εφοδιασμένος με κατάλληλο αριθμό θυρών.

Η δυνατότητα που προσφέρουν οι σερβοκινητήρες στα συστήματα ελέγχου κίνησης με τον παράλληλο έλεγχο ταχύτητας και θέσης με πολύ μεγάλη ακρίβεια και χωρίς περιορισμούς μηχανικής ισχύος (δηλαδή αναγκαίας ροπής) τους καθιστά κατάλληλους για μια τεράστια γκάμα βιομηχανικών εφαρμογών. Το μεγαλύτερο μέρος από αυτές αναφέρεται σε εξελιγμένη αυτοματοποίηση κατασκευαστικών διαδικασιών και σε μεταφορά και συσκευασία υλικών και προϊόντων. Αναλυτικότερα οι εφαρμογές που συναντούν οι σερβοκινητήρες στη βιομηχανία είναι οι ακόλουθες:

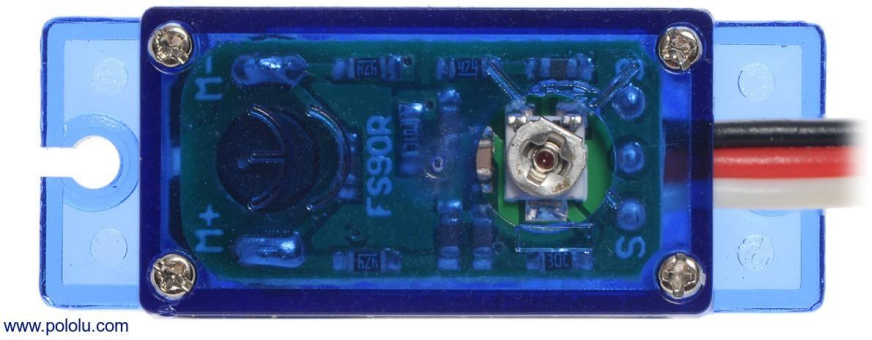
- Σε ρομποτικά συστήματα όλων των ειδών. Τα ρομποτικά συστήματα έχουν την δυνατότητα να υποστηρίζουν μεταξύ άλλων, εξελιγμένες εργαλειομηχανές κατεργασιών υλικών και μηχανές συγκόλλησης και βαφής μετάλλων, αλλά και συστήματα συναρμολόγησης σε κατασκευαστικές βιομηχανίες. Στην κατεργασία των υλικών κατέχουν εξέχοντα ρόλο στην αναβάθμιση των παραδοσιακών εργαλειομηχανών σε εργαλειομηχανές CNC. Χρησιμοποιούνται στις κοπές μετάλλων οποιασδήποτε μορφής. Βρίσκουν ιδιαίτερες εφαρμογές σε τριαξονικά συστήματα κοπής, σε σύγχρονες κοπές εν κινήσει πολλαπλών σταθμών και μπορούν να επιτύχουν κοπές εν κινήσει με συγχρονισμό μέχρι 12 αξόνων. Ιδιαίτερη εφαρμογή συναντούν στις μηχανές συρματουργίας.
- Οι σερβοκινητήρες διευρύνουν την εφαρμογή τους στον κατασκευαστικό τομέα. Πιο συγκεκριμένα, στο βιομηχανικό τομέα χρησιμοποιούνται σε μηχανές που αναλαμβάνουν την επεξεργασία χαρτιού, ξύλου, μαρμάρου. Ιδιαίτερα σημαντικό ρόλο παίζουν στις μηχανές παραγωγής πλαστικών προϊόντων (extruders, μηχανές blow, κ.λ.π), ενώ ειδικές εφαρμογές βρίσκουν σε διάφορες άλλες βιομηχανίες, όπως στην καπνοβιομηχανία και στην τυπογραφία.
- Στον τομέα της μεταφοράς και συσκευασίας υλικών και προϊόντων οι σερβοκινητήρες βρίσκουν μαζική εφαρμογή. Ειδικότερα, χρησιμοποιούνται σε εγκιβωτιστικά συστήματα, σε παλετοποιητικά συστήματα διαφόρων προϊόντων και σε συστήματα pick and place. Ακόμα χρησιμοποιούνται σε μηχανές συσκευασίας, σε καρτονέττες και σε ετικέτες. Τέλος, χρησιμοποιούνται σε γεμιστικά μηχανήματα χύδην, στερεών και υγρών προσόντων.

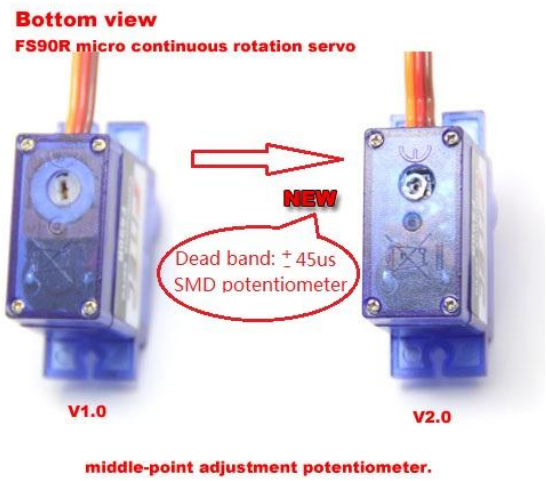
Τα παραπάνω δεδομένα καθίστανται οικεία σε όσους ασχολούνται αποκλειστικά με τους σερβοκινητήρες (επαγγελματίες μηχανικούς), ενώ για την παρούσα εργασία τα πράγματα είναι πιο εύκολα και απλά. Δηλαδή, ο συγκεκριμένος σερβοκινητήρας που αναλύεται είναι μικρός και ελαφρύς αλλά με μεγάλη ισχύ εξόδου και λειτουργεί όπως ακριβώς και τα κανονικά σέρβο απλά είναι σε μικρότερες διαστάσεις (για τις ανάγκες προσομοιώσεις).

Ο μελετώμενος σερβοκινητήρας απευθύνεται σε αρχάριους χρήστες που θέλουν να έχουν ένα σερβομοτέρ με συνεχή κίνηση χωρίς να χρειάζεται να ασχοληθούν με την οικοδόμηση ενός ελεγκτή κινητήρα με ανάδραση και κιβώτιο ταχυτήτων.

Ο σερβοκινητήρας που θα χρησιμοποιήσουμε ονομάζεται FS90R και κατασκευάζεται από την FEETECH και ειδικεύεται στην συνεχόμενη περιστροφή. Στα 6 volt, παρέχεται η μέγιστη ταχύτητα περιστροφή η οποία βρίσκεται γύρω στις 130 RPM (στροφές ανά λεπτό) χωρίς φορτίο και μπορεί να παράγει έως 1.5kg-cm ροπής. Το σέρβο μπορεί να ελεγχθεί χρησιμοποιώντας απευθείας σύνδεση με ένα μικροελεγκτή I/O όπως το ArduinoMega 2560 που θα χρησιμοποιήσουμε για την υλοποίηση του κυκλώματος, χωρίς κάποιο έξτρα ηλεκτρικό εξάρτημα κάτι που το κάνει κορυφαίο σε ρομποτικά projects.

Ο FS90R μετατρέπει τους τυπικούς RCservo παλμούς θέσης σε συνεχόμενη ταχύτητα περιστροφής. Το restpoint του βρίσκεται στα 1.5 ms αλλά μπορεί να ρυθμιστεί με ένα μικρό κατσαβίδι για να γυρίσει το κεντρικά τοποθετημένο ποτενσιόμετρο στο κάτω μέρος του. Παλμοί πλάτους πάνω από του restpoint έχουν ως αποτέλεσμα την αριστερόστροφή περιστροφή του, με ταχύτητα η οποία αυξάνεται όσο το πλάτος του παλμού αυξάνεται. Αντίστροφα παλμοί πλάτους κάτω από του restpoint έχουν ως αποτέλεσμα την δεξιόστροφη κίνηση, με ταχύτητα η οποία αυξάνεται όσο το πλάτος του παλμού μειώνεται.





Εικόνα 3.4 FEETECH FS90R Ποτενσιόμετρο στην κάτω του πλευρά

Το σέρβο μας έχει 250mm απόληξη η οποία φέρει ένα τύπου JRconnector και παρέχει μέσα στην συσκευασία του διάφορες βίδες και βάσεις στήριξης. Η παρακάτω φωτογραφία απεικονίζει τα παρελκόμενα μου υπάρχουν μέσα στην συσκευασία.

Εικόνα 3.5 FEETECH FS90R Παρελκόμενα στην συσκευασία στηρίγματα

Τα χαρακτηριστικά που δίνει το επίσημο site για τον σερβοκινητήρα είναι τα εξής:

Dimensions

Size:	23.2 × 12.5 × 22 mm
Weight:	9 g

General specifications

Digital?:	N
Free-run current @ 6V:	120 mA
Stall current @ 6V:	800 mA[1]
Speed @ 6V:	130 rpm
Stall torque @ 6V:	21 oz·in[1]
Speed @ 4.8V:	100 rpm
Stall torque @ 4.8V:	18 oz·in[1]
Lead length:	10 in
Hardware included?:	Y

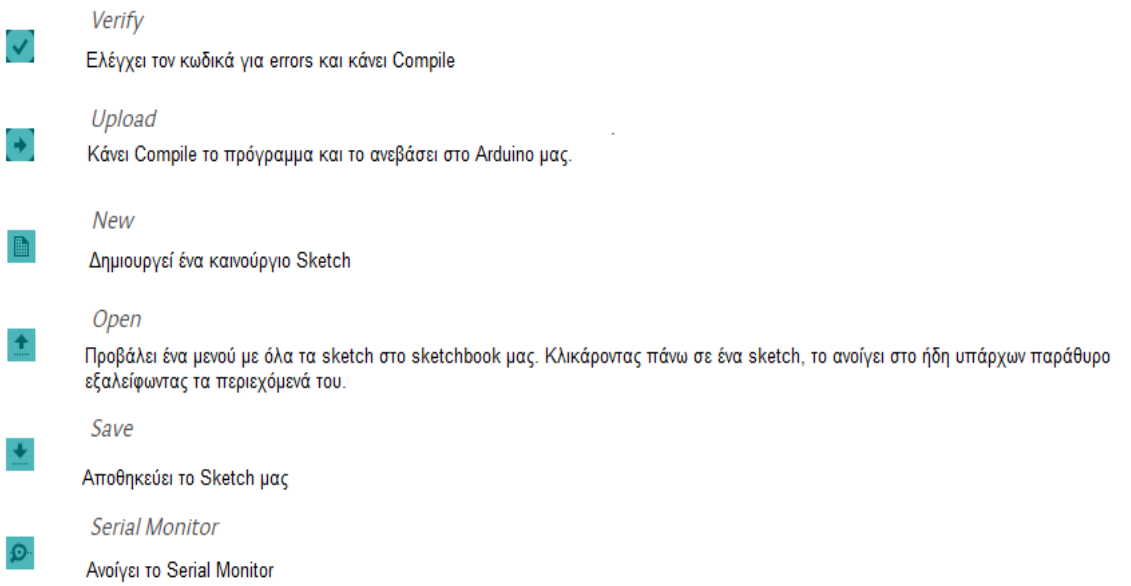
4. ArduinoIDE

Με τα αρχικά IDE συντομογραφία του IntegratedDevelopmentEnvironment αναφερόμαστε στο ενσωματωμένο περιβάλλον ανάπτυξης των μικροελεκτών Arduino. Το ArduinoIDE περιέχει έναν texteditor για να γράφουμε τον κώδικα μας, μια περιοχή μηνυμάτων για να βλέπουμε τις ενέργειες που γίνονται και αν υπάρχουν τυχών βλάβες, μια textconsole, μια γραμμή εργαλείων που περιέχει διάφορα μενού και κουμπιά για τις κύριες λειτουργίες του IDE. Συνδέει τους Arduino και Genuino μικροελεγκτές για να ανεβάσουμε προγράμματα και να επικοινωνήσουν μεταξύ τους.

4.1 Sketches

Τα προγράμματα που γράφονται στο λογισμικό του Arduino (IDE) ονομάζονται **sketches**. Τα sketches αυτά γράφονται στον texteditor και αποθηκεύονται με επέκταση .ino στο τέλος του αρχείου. Ο editor έχει τα χαρακτηριστικά αποκοπής / επικόλλησης και για αναζήτησης/αντικατάσταση κειμένου. Η περιοχή μηνυμάτων μας δείχνει χρήσιμες πληροφορίες καθώς αποθηκεύουμε ή κάνουμε εξαγωγή το sketch μας καθώς και τυχών σφάλματα (errors) που θα προκύψουν. Η κονσόλα απεικονίζει τα σφάλματα και τις λοιπές πληροφορίες από το IDE σε μορφή κειμένου. Η κάτω δεξιά γωνία του παραθύρου απεικονίζει τον μικροελεγκτήArduino που έχουμε επιλέξει καθώς και την σειριακή θύρα που βρίσκεται συνδεδεμένος (SerialPort). Τα κουμπιά της γραμμής εργαλείων μας παρέχουν τις δυνατότητες της επικύρωσης και ανεβάσματος των προγραμμάτων, τις λειτουργίεςcreate, open, savesketches και το άνοιγμα του serialmonitor.

Οι λειτουργίες των κουμπιών του IDE είναι οι εξής :



Επιπλέον εντολές βρίσκονται στα πέντε μενού: **File**, **Edit**, **Sketch**, **Tools**, **Help**. Τα μενού είναι contextsensitive, το οποίο σημαίνει πως μόνο τα χαρακτηριστικά που σχετίζονται με το πρόγραμμα το οποίο φτιάχνουμε είναι διαθέσιμα.

4.2 Μενού File

- **New**
  Δημιουργεί μια νέα φόρμα του Editor στην οποία βρίσκεται η βασική δομή ενός Sketch.

- **Open**
  Μας επιτρέπει να ανοίξουμε ένα παράθυρο πλοήγησης στον υπολογιστή για να βρούμε ένα υπάρχων αρχείο sketch.

- **OpenRecent**
  Μας παρέχει μία μικρή λίστα από τα πιο πρόσφατα Sketches, έτοιμα για άνοιγμα.

- **Sketchbook**
  Μας ανοίγει ένα παράθυρο πλοήγησης μέσα στο φάκελο sketchbook των υπαρχόντων sketches και κλικάροντας πάνω στο κάθε όνομα του sketch το ανοίγει.

- **Examples**
  Κάθε παράδειγμα που παρέχεται από το ArduinoIDE ή κάθε βιβλιοθήκη εμφανίζεται σε αυτό το μενού. Όλα τα παραδείγματα είναι δομημένα σε μορφολογία δέντρου, κάτι το οποίο μας παρέχει εύκολη πρόσβαση ανά αντικείμενο ή βιβλιοθήκη.

- **Close**
  Κλείνει τον editor του IDE τον οποίο επιλέξαμε.

- **Save**
  Αποθηκεύει το Sketch με το ήδη υπάρχον όνομα. Εάν το αρχείο δεν έχει αποθηκευτεί προηγουμένως, ένα όνομα θα του δοθεί στο παράθυρο “Saveas..”

- **Saveas**…
  Μας δίνει την δυνατότητα να αποθηκεύσουμε το ήδη υπάρχων sketch με ένα διαφορετικό όνομα.

- **Pagesetup**
  Μας δείχνει το παράθυρο PageSetup για εκτύπωση του προγράμματος.

- **Print**
  Εκτυπώνει το πρόγραμμα μας σύμφωνα με τις ρυθμίσεις που επιλέξαμε από το παράθυρο PageSetup.

- **Preferences**
  Ανοίξει το παράθυρο προτιμήσεων του IDE όπου μπορούν κάποιες ρυθμίσεις του IDE να παραμετροποιηθούν , όπως πχ. Ή γλώσσα του περιβάλλοντος.

- **Quit**
  Κλείνει όλα τα παράθυρα του IDE. Τα ίδια sketch που ήταν ανοιχτά την στιγμή που πατήσαμε το κουμπί Quit , αυτόματα θα ξανανοίξουν της επόμενη φορά που θα ανοίξουμε το ArduinoIDE.

4.3 Edit

- **Undo/Redo**
  Πηγαίνει ένα η περισσότερα βήματα πίσω καθώς κάναμε edit το sketch μας. Όταν πάμε πίσω με το Undo μπορούμε να ξαναγυρίσουμε μπροστά με το Redo.

- **Cut**
  Αφαιρεί το επιλεγμένο κείμενο από τον editor και το τοποθετεί στο clipboard.

- **Copy**
  Αντιγράφει το επιλεγμένο κείμενο από editor και το τοποθετεί στο clipboard.

- **Copy for Forum**
  Αντιγράφει τον κώδικα του Sketch στο clipboard σε μία φόρμα κατάλληλη για κοινοποίηση σε ένα forum, ολοκληρωμένο μαζί με το syntaxcoloring.

- **Copy as HTML**
  Αντιγράφει τον κώδικα του Sketch στο clipboard σε μορφή HTML, κατάλληλο για εισαγωγή σε ιστοσελίδες.

- **Paste**
  Τοποθετεί τα περιεχόμενα του clipboard στην γραμμή που βρίσκεται ο κέρσορας μέσα στον editor.

- **SelectAll**
  Επιλέγει όλο το περιεχόμενο κειμένου του editor.

- **Comment/Uncomment**
  Τοποθετεί ή αφαιρεί τα // σημάδια σχολίων στην αρχή της κάθε επιλεγμένης γραμμής.

- **Increase/Decrease Indent**
  Προσθέτει ή αφαιρεί ένα διάστημα στην αρχή της κάθε επιλεγμένης γραμμής, μετακινώντας το κείμενο ένα διάστημα προς τα δεξιά ή προς τα αριστερά.

- **Find**
  Ανοίγει το παράθυρο FindandReplace στο οποίο μπορούμε να εισάγουμε το κείμενο που θέλουμε να βρούμε μέσα στον υπάρχων sketch μέσα από διάφορες επιλογές.

- **FindNext**
  Υπογραμμίζει το επόμενο ίδιο κείμενο – εάν υπάρχει- μέσα στο ανοιχτό αρχείο στον texteditor.

- **FindPrevious**
  Υπογραμμίζει το προηγούμενο ίδιο κείμενο – εάν υπάρχει- μέσα στο ανοιχτό αρχείο στον texteditor.

4.4 Sketch

- **Verify/Compile**
  Έλεγχος του sketch για σφάλματα κάνοντας το compile. Θα μας δώσει αναφορά για την μνήμη που χρησιμοποιεί ο κώδικας και οι μεταβλητές στην περιοχή της κονσόλας.

- **Upload**
  Κάνει compile και φορτώνει το binary αρχείο στον προσαρμοσμένο μικροελεγκτή Arduino μας μέσω της αντίστοιχης σειριακής θύρας.

- **UploadUsingProgrammer**
  Αυτή η επιλογή θα παρακάμψει τον bootloader που βρίσκεται στον μικροελεγκτή Arduino. Πρέπει να χρησιμοποιήσουμε την επιλογή Tools>BurnBootloader για να επαναφέρουμε τον bootloader και να μπορέσουμε να κάνουμε πάλι Upload στην USB σειριακή θύρα μας ξανά. Ωστόσο, η επιλογή αυτή μας δίνει την δυνατότητα να χρησιμοποιήσουμε όλη την χωρητικότητα της Flash μνήμης για το sketch μας. Αυτή η επιλογή δεν θα κάψει τις ασφάλειες.

- **ExportCompiledBinary**
  Αποθηκεύει ένα .hex αρχείο το οποίο μπορεί αν αρχειοθετηθεί ή να σταλεί στο Arduino μας χρησιμοποιώντας άλλες μεθόδους.

- **ShowSketchFolder**
  Ανοίγει τον υπάρχων φάκελο sketch.

- **IncludeLibrary**
  Προσθέτει μια βιβλιοθήκη στο sketch μας εισάγοντας την με #include δήλωση στη αρχή του κώδικά μας. Επιπρόσθετα από το μενού αυτό έχουμε πρόσβαση στο LibraryManager και να εισάγουμε νέες βιβλιοθήκες από .zip αρχεία.

- **AddFile**…
  Προσθέτει ένα sourcefile στο sketch μας. (Αντιγράφεται από την υπάρχουσα θέση του).Το καινούριο αυτό αρχείο εμφανίζεται σε μία καινούργια καρτέλα στο sketch παράθυρο. Τα αρχεία μπορούν να αφαιρεθούν από το sketch χρησιμοποιώντας το tabmenu κλικάροντας τι μικρό τριγωνικό εικονίδιο κάτω από serialmonitor στην δεξιά πλευρά του toolbar.

4.5 Tools

- **AutoFormat**
  Αυτόματη μορφοποίηση του κώδικα με σκοπό να είναι όμορφος οπτικά και ευανάγνωστος για τον debugging.

- **ArchiveSketch**
  Δημιουργεί ένα αντίγραφο του υπάρχοντος sketch σε .zip μορφή και αποθηκεύεται στο ίδιο directory που υπάρχει το sketch.

- **FixEncoding**&**Reload**
  Φτιάχνει τις τυχών διαφορές της κωδικοποίησης του editorcharmap με των διαφορετικών λειτουργικών συστημάτων charmaps.

- **SerialMonitor**
  Ανοίγει το serialmonitor παράθυρο και ξεκινάει την ανταλλαγή δεδομένων με το οποιοδήποτε συνδεδεμένο Arduino μικροελεγκτή στην επιλεγμένη θύρα. Αυτό συνήθως επαναφέρει τον μικροελεγκτή στην αρχική του κατάσταση ένα ο μικροελεγκτή υποστηρίζει λειτουργία Reset στο άνοιγμα της serialport.

- **Board**
  Επιλέγουμε το μοντέλο του Arduino μικροελεγκτή που χρησιμοποιούμε.

- **Port**
  Αυτό το μενού περιέχει όλες στις σειριακές συσκευές (Φυσικές ή τεχνητές – RealorVirtual) στον υπολογιστή μας. Συνήθως ανανεώνεται αυτόματα κάθε φορά που ανοίγουμε τα επάνω παράθυρα από το toolsmenu.

- **Programmer**
  Για την επιλογή νέου hardwareprogrammer όταν προγραμματίζουμε ένα μικροελεγκτή ή chip χωρίς να χρησιμοποιούμε την ενσωματωμένη USB σειριακή σύνδεση. Συνήθως δεν μας χρειάζεται αυτή η λειτουργία παρα μόνο ένα κάνουμε burn ένα νέο bootloader σε ένα καινούριο μικροελεγκτή.

- **BurnBootloader**
  Τα στοιχεία σε αυτό το μενού μας παρέχουν την δυνατότητα να γράψουμε έναν bootloader μέσα στον μικροελεγκτή ενός Arduino. Αυτό δεν χρειάζεται για την κανονική χρήση νέος Arduino ή Genuino αλλά είναι χρήσιμο ένα αγοράσουμε έναν καινούργιο ATmega μικροελεγκτή (οι οποίοι συνήθως έρχονται χωρίς προ εγκατεστημένο bootloader). Σιγουρευτείτε ότι έχετε επιλέξει το σωστό board από το μενού Boards προτού γράψετε ένα bootloader στο αντίστοιχο board. Ακόμη, η επιλογή αυτή θα ρυθμίσει τις σωστές ασφάλειες.

4.6 Help

Σε αυτό το μενού μπορούμε να βρούμε εύκολα δεκάδες έγγραφα που παρέχονται με την εγκατάσταση του ArduinoIDE. Έχουμε πρόσβαση στους οδηγούς GettingStarted , Reference του IDE και σε άλλα έγγραφα χωρίς σύνδεση στο διαδίκτυο.

- **Find In Reference**
  Η μόνη επιλογή στο μενού αυτό. Επιλέγει απευθείας την συσχετισμένη σελίδα από το τοπικό αντίγραφο της Reference για την λειτουργία ή εντολή κάτω από την θέση του κέρσορα.

4.7 Sketchbook

Το ArduinoIDE χρησιμοποιεί ένα πρότυπο μοντέλο sketchbook: Ένα προκαθορισμένο φάκελο που μπορείς να αποθηκεύσεις τα προγράμματα ή sketches. Τα sketches που έχουμε δημιουργήσει μπορούν να ανοιχτού από το μενού **File>Sketchbook** ή από το κουμπί **Open** στην γραμμή εργαλείων. Την πρώτη φορά που θα χρησιμοποιήσουμε το λογισμικό αυτό, θα δημιουργήσει αυτόματα αυτό το φάκελο και θα τον καθορίσει ως το Sketchbook μας. Μπορούμε να βρούμε ή ακόμη και να αλλάξουμε την διαδρομή/τοποθεσία του sketchbook μας από την επιλογή **Preferences** στο IDE.

4.8 Uploading

Προτού φορτώσουμε (Upload) το sketch μας στο Arduino, πρέπει να έχουμε επιλέξει το σωστό Arduino και την σωστή σειριακή θύρα στα μενού **Tools>Board**και **Tools>Port** αντίστοιχα. Στα Mac , η σειριακήθύρα έχει συνήθως την μορφή /dev/tty.usbmodem241 (για το Uno ή Mega2560 ή το Leonardo) ή /dev/tty.usbserial-1B1 (για το Duemilanove ή κάποιο παλαιότερο USBboard), ή /dev/tty.USA19QW1b1P1.1 (για κάποια σειριακή πλακέτα (serialboard) συνδεδεμένη με έναν KeyspanUSB-to-Serial αντάπτορα). Στα Windows στα οποία θα υλοποιηθεί η εργασία μας συνήθως έχει την μορφή COM1 ή COM2 (για κάποια σειριακή πλακέτα) ή COM 4, COM5, COM7, κλπ (για πλακέτα με σύνδεση USB).Για να βρούμε την θύρα αυτή μπορούμε να ψάξουμε για μια USBserialdevice μέσα στο DeviceManager των Windows.Στα Linux έχει την μορφή /dev/ttyACMx , /dev/ttyUSBx ή παρόμοια.

Μόλις επιλέξουμε την σωστή σειριακή θύρα και πλακέτα Arduino, πατάμε το κουμπί **upload** στη **γραμμήεργαλειών** ή το κουμπί**Upload** από το **Sketchμενού**. Με την διαδικασία αυτή το Arduino μας θα κάνει reset και θα ξεκινήσει η διαδικασία φόρτωσης του κώδικα. Με Arduino παλαιότερα του Diecimila που δεν έχουν την δυνατότητα auto-reset (αυτόματης επαναφοράς) θα πρέπει να χρησιμοποιήσουμε το κουμπί reset στο Arduino αμέσως πριν ξεκινήσει η φόρτωση του κώδικα. Στα περισσότερα Arduino θα δούμε τα LEDRX και TX να αναβοσβήνουν καθώς τους φορτώνουμε τον κώδικα του sketch.

Το ArduinoSoftware (IDE) θα μας εμφανίσει ένα μήνυμα όταν η φόρτωση (upload) του sketch ολοκληρωθεί η αλλιώς ένα μήνυμα σφάλματος.

Όταν φορτώνουμε ένα sketch, χρησιμοποιούμε τον Arduinobootloader , ένα μικρό προεγκατεστημένο πρόγραμμα στον μικροελεγκτή του Arduino. Αυτό μας παρέχει την δυνατότητα να φορτώσουμε τον κώδικα μας χωρίς να χρειαστούν επιπλέον εργαλεία. Ο bootloader είναι ενεργός για μερικά δευτερόλεπτα καθώς το Arduino κάνει reset, μετά ξεκινάει με όποιο ήταν το τελευταίο Sketch το οποίο φορτώθηκε στον μικροελεγκτή. Obootloader θα αναβοσβήσει το pin 13 LED κάθε φορά που θα ξεκινήσει την λειτουργία του όπως πχ κατά την διαδικασία του reset.

4.9 Libraries (Βιβλιοθήκες)

Οι βιβλιοθήκες μάς παρέχουν επιπλέον λειτουργίες για τα sketch μας, όπως πχ. Να χρησιμοποιούμε διάφορα ηλεκτρονικά στοιχεία ή να χρησιμοποιούμε δεδομένα. Για να χρησιμοποιήσουμε μια βιβλιοθήκη σε ένα sketch , μπορούμε να την επιλέξουμε από το μενού **Sketch>ImportLibrary** . Με τον τρόπο αυτό μπορούμε να εισάγουμε μια ή περισσότερες δηλώσεις **#include** στο πάνω μέρος του sketch μας και να προσθέσουμε την βιβλιοθήκη αυτή στο sketch.

Καθώς οι βιβλιοθήκες φορτώνονται στο Arduino μέσω του sketch , μεγαλώνουν τον χώρο που καταλαμβάνει το sketch στο Arduino. Ένα δεν μας χρειάζεται πια μία βιβλιοθήκη, απλά διαγράφουμε την **#include** δήλωσή της στο πάνω μέρος του κώδικα του Sketch μας.

Υπάρχει μια μεγάλη λίστα βιβλιοθηκών. Κάποιες βιβλιοθήκες είναι προ εγκατεστημένες στο λογισμικό του Arduino. Άλλες μπορούν να κατεβαστούν από διάφορες πηγές μέσω του **LibraryManager**. Με βάση την έκδοση 1.0.5 του IDE μπορούμε να εισάγουμε μια βιβλιοθήκη ακόμη και από ένα zip αρχείο και να την χρησιμοποιήσουμε σε κάποιο sketch . Ακόμη μπορούμε να συγγράψουμε την δικιά μας βιβλιοθήκη.

4.10 Third-Party Hardware

Η υποστήριξη υλικών τρίτων κατασκευαστών μπορεί να προστεθεί στο hardwaredirectory του sketchbook. Εγκατεστημένες πλατφόρμες οι οποίες μπορεί να περιέχουν ορισμούς συσκευών (οι οποίες εμφανίζονται στο μενού board), βιβλιοθήκες πυρήνα, bootloaders και προγραμματιστικούς ορισμούς.

Για την εγκαταστάσεις τέτοιου είδους, δημιουργούμε ένα **hardware**directory, μετά κάνουμε εξαγωγή (unzip) την πλατφόρμα τρίτων κατασκευαστών στο δικό της υποκατάλογο (sub-directory).

Ποτέ δεν χρησιμοποιούμε το όνομα “arduino” ως το όνομα του υποκαταλόγου αλλιώς θα γράψουμε πάνω στην προεγκατεστημένη πλατφόρμα του Arduino.Για να απεγκαταστήσουμε μια πλατφόρμα απλά διαγράφουμε τον υποκατάλογο της.

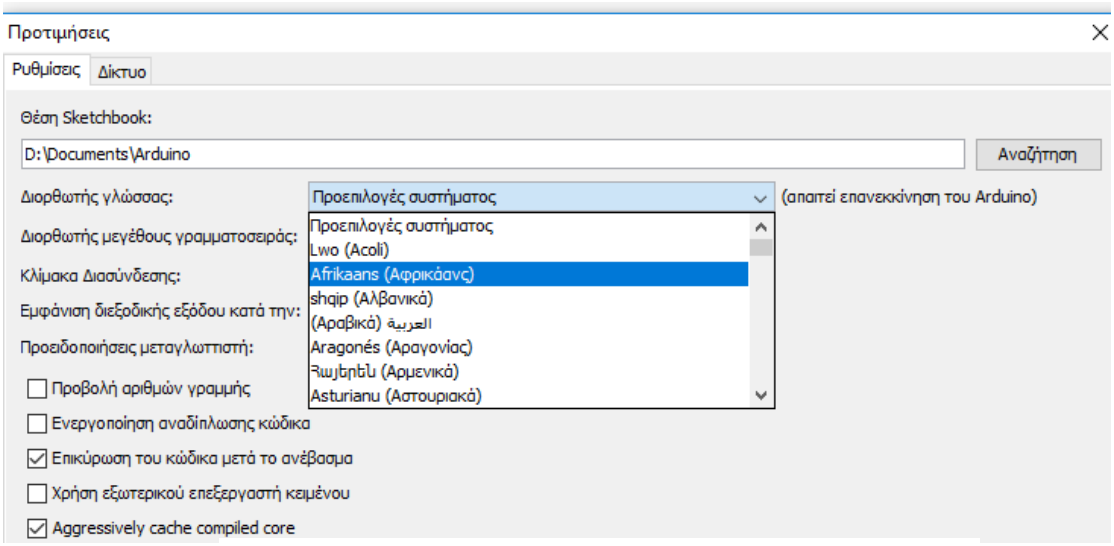
4.11 SerialMonitor

ToSerialMonitor απεικονίζει την σειριακή αποστολή δεδομένων από το Arduino μέσω σειριακής επικοινωνίας ή USB. Για να στείλουμε δεδομένα στο Arduino , γράφουμε το κείμενο που θέλουμε και πατάμε το κουμπί “send” ή το enter του πληκτρολογίου. Επιλέγουμε τον ρυθμό baud από το drop-down μενού που αντιστοιχεί στο επιθυμητό ρυθμό που έχουμε επιλέξει στο Serial.begin του sketch μας. Είναι καλό να γνωρίζουμε πως σε όλα τα λειτουργικά συστήματα Windows, Mac ή Linux το Arduino θα κάνει reset καθώς συνδέεται με το serialmonitor. Το SerialMonitor δεν επεξεργάζεται χαρακτήρες ελέγχου και εάν το sketch μας χρειάζεται πλήρες έλεγχο για σειριακή επικοινωνία με χαρακτήρες ελέγχου, μπορούμε να χρησιμοποιήσουμε ένα εξωτερικό τερματικό και να το συνδέσουμε με την COMθύρα που είναι συνδεμένο το Arduino μας.

4.12 Preferences (προτιμήσεις)

Κάποιες προτιμήσεις μπορούν να ρυθμιστούν από το μενού Preferences όπου βρίσκεται κάτω από το μενού **Arduino** στα Mac ή το μενού **File** στα Windows ή Linux. Οι υπόλοιπες μπορούν να βρεθούν στο αρχείο preferences που ή τοποθεσία του απεικονίζεται στο preferencedialog.

4.13 Language Support (Υποστηριζόμενες Γλώσσες)



Εικόνα 4.1 Μενού Προτιμήσεις για αλλαγή γλώσσας

Από την έκδοση 1.0.1, το ArduinoIDE έχει μεταφραστεί σε πάνω από 30 διαφορετικές γλώσσες. Η προκαθορισμένη γλώσσα που φορτώνει το IDE είναι ή γλώσσα του λειτουργικού μας συστήματος. (Σημείωση: Στα Windows και Linux , η γλώσσα αυτή καθορίζεται από τις τοπικές ρυθμίσεις που ελέγχουν τις τοποθεσίες και ημερομηνίες και όχι από την γλώσσα η οποία έχει εγκατασταθεί να απεικονίζεται το λειτουργικό σύστημα).

Ένα θέλουμε χειροκίνητα να αλλάξουμε την γλώσσα, ανοίγουμε το ArduinoSoftware (IDE) και ανοίγουμε το παράθυρο Preferences. Μετά στον διορθωτή γλώσσας υπάρχει ένα dropdown μενού των υποστηριζόμενων γλωσσών. Επιλέγουμε την γλώσσα τις προτίμησης μας από το μενού και επανεκκινούμε το λογισμικό για να χρησιμοποιήσουμε την γλώσσα αυτή. Εάν η γλώσσα του συστήματος μας δεν υποστηρίζεται από το ArduinoIDE τότε η προεπιλεγμένη γλώσσα α είναι τα αγγλικά.

Μπορούμε να επαναφέρουμε το λογισμικό στην προεπιλεγμένη γλώσσα ή να επιλέξουμε την γλώσσα σύμφωνα με αυτήν του λειτουργικού μας συστήματος, επιλέγοντας SystemDefault από το drop-down μενού του**EditorLanguage**. Αυτή η αλλαγή θα γίνει μόλις επανεκκινήσουμε το IDE. Ομοίως μόλις αλλάξουμε την γλώσσα του συστήματος μας πρέπει να επανεκκινήσουμε το IDE για να αναβαθμιστεί με την νέα προεπιλεγμένη γλώσσα του συστήματος.

4.14 Boards (Πλακέτες)

Η επιλογή πλακέτας έχει δύο ρυθμίσεις:

- Ρυθμίζει τις παραμέτρους (π.χ. την ταχύτητα του επεξεργαστή και τον ρυθμό baud (baudrate)) που χρησιμοποιούνται όταν κάνουμε compile και φορτώνουμε τα sketch μας στο Arduino
- Ρυθμίζει το αρχείο και τις φίσες που θα χρησιμοποιηθούν από την εντολή **burnbootloader** (φόρτωση/γράψιμο bootloader)

Το ArduinoSoftware (IDE) περιέχει προεγκατεστημένα υποστήριξη για τις πλακέτες που θα αναφερθούν στην παρακάτω λίστα, βασισμένη στον κώδικα AVR .Ο **BoardsManager** που βρίσκεται στην κανονική εγκατάσταση του IDE έχει την δυνατότητα πρόσθεσης νέων πλακετών βασισμένες σε εκδόσεις όπως το ArduinoDue, Zero, Edison, Galileo και λοιπά.

Arduino Yùn	*Arduino Ethernet*
Arduino/Genuino Uno	*Arduino Fio*
Arduino Diecimila or Duemilanove w/ ATmega168	*Arduino BT w/ ATmega328P*
Arduino Nano w/ ATmega328P	*LilyPad Arduino USB*
Arduino/Genuino Mega 2560	*LilyPad Arduino*
Arduino Mega	*Arduino Pro or Pro Mini (5V, 16 MHz) w/ ATmega328P*
Arduino Mega ADK	*Arduino NG or older w/ ATmega168*
Arduino Leonardo	*Arduino Robot Control*
Arduino/Genuino Micro	*Arduino Robot Motor*
Arduino Esplora	*Arduino Gemma*
Arduino Mini w/ ATmega328P	*Arduino Ethernet*

5. Arduino και Processing

H**Processing**είναι μια γλώσσα ανοιχτού κώδικα / εργαλείο ανάπτυξης για γράψιμο προγραμμάτων σε άλλους υπολογιστές. Πολύ χρήσιμη όταν οι υπολογιστές αυτοί θέλουν να επικοινωνούν με το Arduino όπως π.χ. να απεικονίζουν ή να αποθηκεύουν κάποια δεδομένα τα οποία συλλέγονται από το Arduino.

Το περιβάλλον εργασίας της Processing (PDE – ProcessingDevelopmentEnvironment) είναι παρόμοιο με το περιβάλλον του ArduinoIDE, δηλαδή περιέχει ένα texteditor, μια περιοχή μηνυμάτων, κονσόλα για τις διαδικασίες και τα τυχών σφάλματα και στο πάνω μέρος μια γραμμή εργαλειών με τις εξής λειτουργίες και καρτέλες.

Εικόνα 5.1 Οι 2 εκδόσεις βιβλίων για εκμάθηση της Processing

File

- ***New***
  Δημιουργία ενός νέου sketch σε ένα νέο παράθυρο,ονομαζόμενο με την ημερομηνία δημιουργίας του στην μορφή: "sketch_YYMMDDa".
- ***Open****...*
  Άνοιγμα ενός sketch σε ένα νέο παράθυρο.
- ***OpenRecent***
  Άνοιγμαενός από τα sketchτα οποία έκλεισαν πρόσφατα.
- ***Sketchbook...***
  Άνοιγμα ενός νέου παραθύρου με την λίστα όλων των sketch που βρίσκονται στο sketchbook.
- ***Examples...***
  Άνοιγμα ενός νέου παραθύρου με την λίστα όλων των παραδειγμάτων.

- ***Close***
  Κλείσιμο του sketchπου βρίσκεται πάνω στην επιφάνεια του προγράμματος. Ένα είναι το μόνοsketchτο οποίο είναι ανοιχτό θα μας ερωτηθεί εάν θέλουμε να βγούμε από το πρόγραμμα. Εάν θέλουμε να βγούμε από το πρόγραμμα χωρίς αυτό το μήνυμα πατάμε Quitαντί για Closeγια έξοδο.
- ***Save***
  Αποθήκευση του ανοιχτού sketch στην παρούσα του μορφή.
- ***Saveas...***
  Αποθήκευση του ανοιχτού sketch, με την επιλογή μετονομασίας του. Εάν αλλάξουμε το όνομα δεν αλλάζει την μορφή του προηγούμενου αποθηκευμένου sketch.
- ***Export***
  Εξάγει μία εφαρμογή Java ως εκτελέσιμο αρχείο και ανοίγει το φάκελο που περιέχει τα αρχεία που έχουν εξαχθεί.
- ***PageSetup***
  Ρυθμίσεις σελίδας για εκτύπωση.
- ***Print (Ctrl+P)***
  Εκτυπώνει τον κώδικα μέσα στον texteditor.
- ***Preferences***
  Αλλάζει κάποιους από τους τρόπους που λειτουργεί η Processing. (Το στοιχείο αυτό βρίσκεται στο μενούProcessingστοMacOSX.)
- ***Quit***
  Έξοδος από το περιβάλλον της Processingκαι κλείσιμο όλων των Processingπαραθύρων. (Το στοιχείο αυτό βρίσκεται στο μενού Processing στα MacOSX.)

Edit

- ***Undo***
  Αναστρέφει την τελευταία εντολή η αλλαγή που πληκτρολογήθηκε. Ακύρωση της εντολής Undoεπιλέγοντας Edit » Redo.
- ***Redo***
  Αναστρέφει την τελευταία εντολήUndo. Αυτή η επιλογή είναι διαθέσιμη μόνο εάν έχει προηγηθεί εντολή Undo.
- ***Cut***
  Αφαιρεί και αντιγράφει το επιλεγμένο κείμενο στο clipboard (έναoff-screentextbuffer).
- ***Copy***
  Αντιγράφει το επιλεγμένο κείμενο στοclipboard.

- ***CopyasHTML***
  Αντιγράφει τον κώδικα του Sketch στο clipboard σε μορφή HTML,ακριβώς όπως εμφανίζεται στο περιβάλλον της Processing, κατάλληλο για εισαγωγή σε ιστοσελίδες.
- ***Paste***
  Τοποθετεί τα περιεχόμενα του clipboard στην γραμμή που βρίσκεται ο κέρσορας μέσα στον editor.
- ***SelectAll***
  Επιλέγει όλο το περιεχόμενο κειμένου του editor.
- ***AutoFormat***
  Προσπαθεί να μορφοποιήσει τον κώδικα ώστε να είναι πιο ευανάγνωστος στο ανθρώπινο μάτι. Παλαιότεραονομαζόταν*Beautify*.
- ***Comment/Uncomment***
  Τοποθετεί ή αφαιρεί τα // σημάδια σχολίων στην αρχή της κάθε επιλεγμένης γραμμής.
- ***IncreaseIndent***
  Μεταφέρει της εσοχή του κειμένου 2 διαστήματα προς τα μέσα.
- ***DecreaseIndent (Ctrl+[)***
  Αφαιρεί την εσοχή που έχει γίνει με την λειτουργίαIncreaseIndent.
- ***Find...***
  Ανοίγει το παράθυρο FindandReplace στο οποίο μπορούμε να εισάγουμε το κείμενο που θέλουμε να βρούμε μέσα στον υπάρχων sketch μέσα από διάφορες λειτουργίες
- ***FindNext***
  Υπογραμμίζει το επόμενο ίδιο κείμενο – εάν υπάρχει- μέσα στο ανοιχτό αρχείο στον texteditor.
- ***FindPrevious***
  Υπογραμμίζει το προηγούμενο ίδιο κείμενο – εάν υπάρχει- μέσα στο ανοιχτό αρχείο στον texteditor.
- ***UseSelectionforFind***
  Ορίζει το επιλεγμένο κείμενο ως στοιχείο για να βρεθεί από της λειτουργίες FindNext και FindPrevious.

Sketch

- ***Run***
  Εκτελεί τον κώδικα (Κάνε compile τον κώδικα, ανοίγει το παράθυρο απεικόνισης, και τρέχει μέσα το sketch)

- ***Present***
  Εκτελεί τον κώδικα στο κέντρο της οθόνης με ένα μονόχρωμο φόντο. Κάνουμε κλικ στο κουμπί "stop" κάτω αριστερά για να βγούμε από την παρουσίαση ή πατάμε το πλήκτρο Escape. Στη καρτέλα Preferences αλλάζουμε το χρώμα του φόντου.

- ***Tweak***
  Εκτελεί τον κώδικα με ένα τρόπο όπου κάποιες χρωματικές και μεταβλητές τιμές μπορούν να αλλάξουν καθώς ο κώδικας συνεχίζει να εκτελείτε. Tosketchπρέπει να αποθηκευτεί πριν εκτελεστεί ως έναsketchtoTweak.

- ***Stop***
  Εάν ο κώδικας εκτελείτε, σταματάει την εκτέλεση του. Προγράμματα που γράφτηκαν χωρίς την λειτουργία draw() σταματάνε αυτόματα μόλις ολοκληρωθεί η λειτουργία draw.

- ***ImportLibrary***
  Προσθέτει τις απαραίτητες δηλώσεις στο πάνω μέρος του sketch. Για παράδειγμα επιλέγοντας Sketch » ImportLibrary » pdfπροσθέτει μια δήλωση "importprocessing.pdf.*;" στο πάνω μέρος του αρχείου. Αυτές οι δηλώσει είναι απαραίτητες για την χρήση βιβλιοθηκών. ΕπιλέγουμεAddLibraries... για να ανοίξουμε τονLibraryManagerγια να βρούμε και να εγκαταστήσουμε καινούριες βιβλιοθήκες.

- ***ShowSketchFolder***
  Ανοίγει τον φάκελο του υπάρχοντος sketch.

- ***AddFile...***
  Ανοίγει το παράθυρο περιήγησης. Επιλέγουμε εικόνα, γραμματοσειρά ή άλλα αρχεία πολυμέσων για να τα προσθέσουμε στον φάκελο δεδομένων "data" του sketch.

Debug

- ***Enable Debugger***
  Ενεργοποιεί τον debugger. Παρατηρούμε ότι το κουμπί Run αλλάζει σε Debug. Νέα κουμπιά όπως το Continue και tep εμφανίζονται, μαζί με ένα ξεχωριστό παράθυρο για να βλέπουμε τις τιμές των μεταβλητών.
- ***Continue***
  Εκτελεί τον κώδικα μέχρι το επόμενο breakpoint.
- ***Step***
  Εκτελεί τον κώδικα μια γραμμή την φορά. (Παρατηρούμε πως όταν ο κώδικας φτάσει στο τέλος μιας κλήσης λειτουργίας, ο debuggerθα το αλλάξει σε "continue.")
- ***StepInto***
  Προχωράει τον debugger μέσα στο εσωτερικό μιας κλήσης λειτουργίας. Αυτό λειτουργεί μόνο για λειτουργίες ορισμένες από τον χρήστη μέσα στοsketch.
- ***StepOut***
  Προχωράει τον debugger στο εξωτερικό μιας κλήσης λειτουργίας. Αυτό λειτουργεί μόνο για λειτουργίες ορισμένες από τον χρήστη μέσα στο sketch.
- ***ToggleBreakpoint***
  Προσθέτει ή αφαιρεί ένα breakpoint. Όταν προστεθεί ένα breakpoint, ο αριθμός τις γραμμής αλλάζει με το σύμβολο <>.

Tools

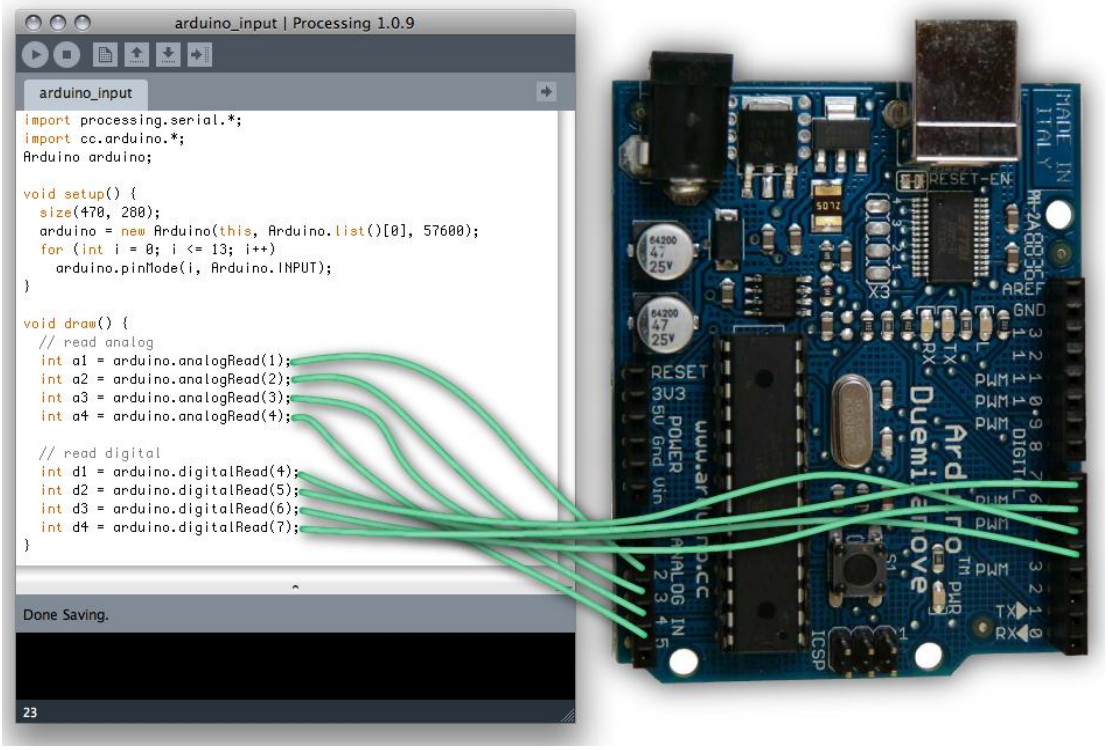
- ***CreateFont...***
  Μετατρέπει τις γραμματοσειρές, σε γραμματοσειρές τύπου Processing (VLW) και τις προσθέτει στο τρέχων sketch. Ανοίγει ένα κουτί διαλόγου (dialogbox) που παρέχει ρυθμίσεις για την γραμματοσειρά, το μέγεθος, την ευκρίνεια κλπ. Το μέγεθος της μνήμης που χρειάζεται για την γραμματοσειρά, καθορίζεται από το επιλεγμένο μέγεθος και τον αριθμό των χαρακτήρων που επιλέχτηκαν από το μενού "Characters...". Οι γραμματοσειρές μπορούν επίσης να δημιουργηθούν μέσα στον κώδικα, με την λειτουργία createFont().
- ***ColorSelector...***
  Περιβάλλον για την επιλογή χρωμάτων. Για κάθε χρώμα οιHSB, RBG, καιHεχτιμές απεικονίζονται. Η Hex τιμή μπορεί να αντιγραφεί στο clipboardμε το κουμπίCopy.
- ***ArchiveSketch***
  Δημιουργεί μία αντιγραφή του τρέχοντος sketch σε .zip μορφή. Αυτό το αρχείο τοποθετείτε στον ίδιο φάκελο με το sketch.

- ***Install "processing-java"***
  Εγκαθιστά ένα πρόγραμμα processing-java για να καθιστά εφικτή την δημιουργία και εκτέλεση Javamode sketches από την γραμμή εντολών.
- ***MovieMaker***
  Δημιουργεί μια ταινία QuickTime από μια αλληλουχία εικόνων.Περιέχονται επιλογές όπως ρύθμιση του μεγέθους τις εικόνας το καρέ, την συμπίεση όπως και το αρχείο ήχου.
- ***AddTool...***
  Ανοίγει τον διαχειριστή εργαλείων (ToolManager) για να βρούμε και να εγκαταστήσουμε καινούρια εργαλεία.

Help

- ***Environment***
  Ανοίγει την ιστοσελίδα αναφοράς του ProcessingDevelopmentEnvironment (αυτήν την σελίδα) στονwebbrowser.
- ***Reference***
  Ανοίγει την επιλογή reference στον web browser μας. Περιέχει αναφορές για την γλώσσα, το περιβάλλον προγραμματισμού και τις βιβλιοθήκες πυρήνα.
- ***FindinReference***
  Επιλογή ενός στοιχείου από την γλώσσα Processing στον text editor και επιλέγουμε Find inReference για να ανοίξουμε αυτήν την σελίδα στον webbrowser μας.
- ***LibrariesReference***
  Επιλογή από την λίστα για άνοιγμα της αναφοράς για τις συμβατές βιβλιοθήκες.
- ***ToolsReference***
  Επιλογή από την λίστα για άνοιγμα της αναφοράς για τα συμβατάεργαλεία.
- ***Getting Started***
  Ανοίγειμία online Getting Started tutorial σελίδα στον web browser.
- ***Troubleshooting***
  Ανοίγειμία online Troubleshooting wiki σελίδα στον web browser.
- ***Frequently Asked Questions***
  Ανοίγειμία online FAQ wiki σελίδα στον web browser.
- ***The Processing Foundation***
  Ανοίγει την σελίδα του Foundation στον web browser.
- ***Visit Processing.org***
  Ανοίγει την αρχική σελίδα του ιστότοποτης Processing στον web browser.

Το ArduinoSoftware (IDE) συμπεριλαμβάνει κάποια βασικά παραδείγματα για επικοινωνία με την Processing (στην καρτέλα **Examples**>**Communication**). Αυτά τα παραδείγματα μας χρησιμεύουν όταν θέλουμε να γράψουμε προγράμματα και Arduino και Processing τα οποία πρέπει να επικοινωνούν μεταξύ τους. Εάν απλά θέλουμε να ελέγχουμε ένα Arduino από ένα πρόγραμμα Processing μπορούμε να συμπεριλάβουμε την Arduino Βιβλιοθήκη που θα εξηγήσουμε παρακάτω.



Εικόνα 5.2 Αντιστοίχιση λειτουργιών Processingμε τα pintου Arduino

5.1 Arduino βιβλιοθήκηγια Processing (και Firmata)

Με την βιβλιοθήκη αυτή, μπορούμε να ελέγξουμε το Arduino μας από την Processing χωρίς να γράψουμε κώδικα στο ArduinoIDE. Αντιθέτως, μπορούμε να ανεβάσουμε ένα πρόγραμμα (standardfirmware) στην πλακέτα μας και να επικοινωνούμε χρησιμοποιώντας αυτή την βιβλιοθήκη. Tofirmware αυτό ονομάζεται Firmata και εμπεριέχετε στο λογισμικό του Arduino.Στον ιστότοπο του Arduino μπορούμε να κατεβάσουμε την συγκεκριμένη βιβλιοθήκη και να την εγκαταστήσουμε ως εξής:

Οδηγίες Εγκατάστασης:

1. Αρχικά εξάγουμε (unzip) την βιβλιοθήκη και αντιγράφουμε τον φάκελο “arduino” στον υποφάκελο “libraries” στο ProcessingSketchbook μας.(Η τοποθεσία του οποίου μπορεί να βρεθεί ανοίγοντας το μενού ProcessingPreferences. Εάν δεν έχουμε υποφάκελο “libraries” δημιουργούμε έναν).
2. Τρέχουμε το ArduinoIDE, ανοίγουμε τα παραδείγματα Examples>Firmata>StandardFirmatasketch και το φορτώνουμε (upload) στο Arduino.
3. Παραμετροποιούμε την Processing για την σειριακή επικοινωνία, με βάση τις πληροφορίες που μας δίνονται στο http://processing.org/reference/libraries/serial/
4. Μέσα στην Processing, ανοίγουμε ένα από τα παραδείγματα που βρίσκεται μέσα στην βιβλιοθήκη (**ArduinoLibrary**).
5. Επεξεργαζόμαστε τον κώδικα του παραδείγματος ώστε να επιλέξουμε την σειριακή θύρα που χρησιμοποιείται από το Arduino. Συγκεκριμένα αλλάζουμε το [0] στην γραμμή : a. arduino = new Arduino(this, Arduino.list()[0], 57600);
6. Τρέχουμε το παράδειγμα

Αναφορικά οι λειτουργίες αυτές βρίσκονται στην βιβλιοθήκη της Processing (**ArduinoLibrary**) και γίνεται επικοινωνία με το Arduino μέσω της Processing , την στιγμή που το Firmatasketch έχει εγκατασταθεί.

- **Arduino.list():**
  Επιστρέφει την λίστα με όλες τις διαθέσιμεςσειριακές συσκευές. Εάν η πλακέτα Arduino μας είναι συνδεδεμένη στον υπολογιστή, όταν καλούμε αυτή, η συσκευή μας θα βρίσκεται στην λίστα.

- **Arduino(parent, name, rate):**
  Δημιουργεί ένα Arduinoobject. Ο Γονιός (**parent**) πρέπει να γράφει “αυτός” χωρίς τις παρενθέσεις, το όνομα (**name**) πρέπει να είναι το όνομα της σειριακής συσκευής και ο ρυθμός (**rate**) είναι η ταχύτητα της σύνδεσης (συνήθως 57600). Στην δεύτερη έκδοση της βιβλιοθήκης (v2), η ταχύτητα σύνδεσης είναι προαιρετική.

- **pinMode(pin, mode):**
  Ορίζεταιέναψηφιακό pin ωςείσοδος, έξοδος, λειτουργία servo (Arduino.INPUT, Arduino.OUTPUT, ή Arduino.SERVO).

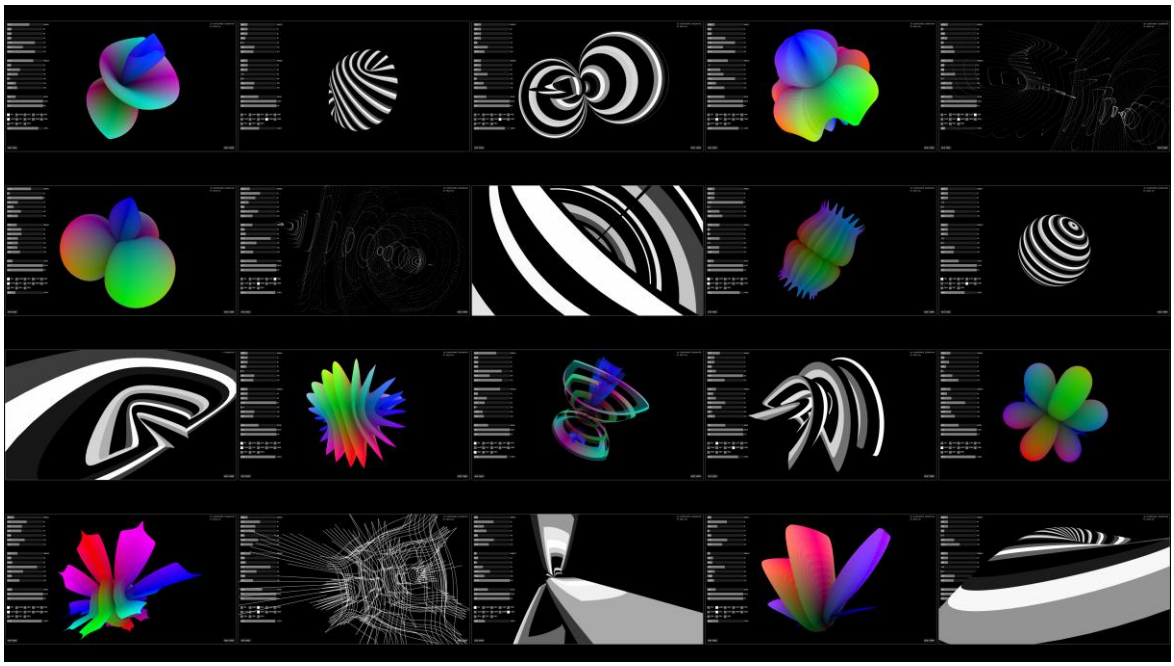
- **digitalRead(pin):**
  Επιστρέφει την τιμή ενός ψηφιακού pin , το οποίο είναι είτε Arduino.LOW είτε Arduino.HIGH. (Το pin πρέπει να έχει οριστεί ώς είσοδος).

- **digitalWrite(pin, value):**
  Γράφει Arduino.LOW ή Arduino.HIGH σεέναψηφιακό pin.

- **analogRead(pin):**
  Επιστρέφει την τιμή μία αναλογικής εισόδου (από 0 έως 1023).

- **analogWrite(pin, value):**
  Γράφει μια αναλογική τιμή (PWM παλμό) σέ ένα ψηφιακό pin το οποίο τον υποστηρίζει (pin 3,5,6,9,10,11). Η τιμή θα πρέπει να είναι από μόνιμα κλειστό 0 έως μόνιμα ανοικτό 180.

- **servoWrite(pin, value):**
  Γράφει μία τιμή σέ έναν σερβοκινητήρα η οποία θα βρίσκεται μεταξύ του 0 και του 180.



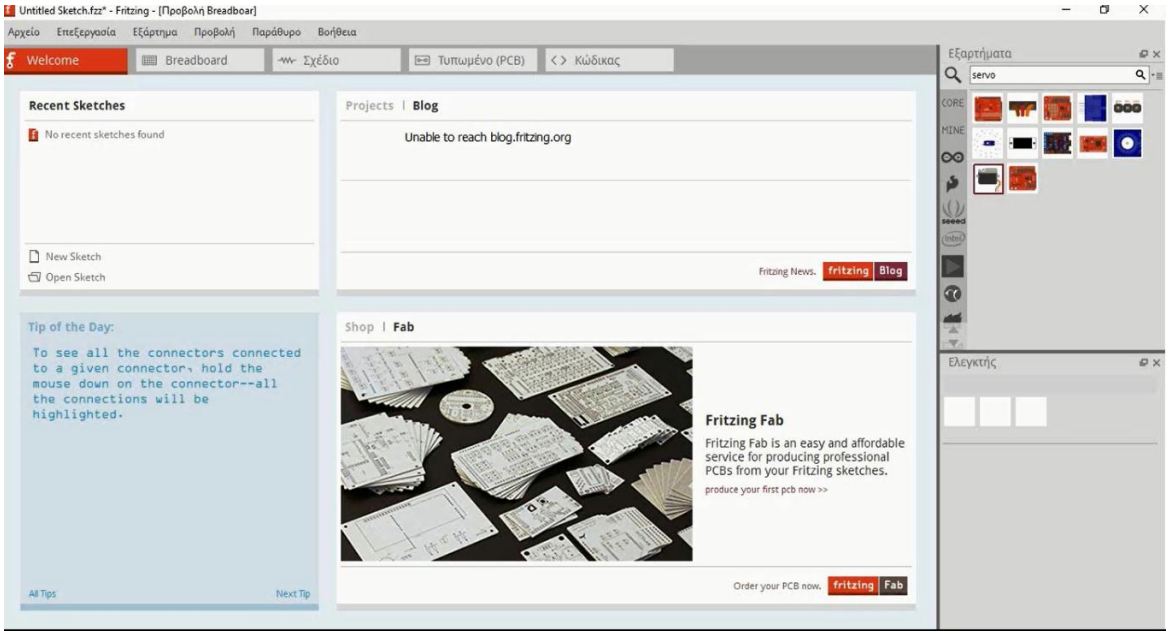
Εικόνα 5.3Supershapes δημιουργημένα με Processing

6. Δημιουργία κυκλώματος

Αρχικά η συνδεσμολογία του κυκλώματος μας θα σχεδιαστεί με το λειτουργικό Fritzing.Το Fritzing είναι ένα δωρεάν εργαλείο σχεδίασης κυκλωμάτων. Είναι πολύ απλό στην χρήση του και έχει ενσωματωμένα όλα τα ηλεκτρονικά εξαρτήματα που θα χρειαστείτε για την σχεδίαση των κυκλωμάτων σας. Μέσα σε αυτό θα βρούμε και όλες τις πλακέτες Arduino καθώς και διάφορα αισθητήρια γι' αυτό. Το Fritzing διαθέτει μια μεγάλη παγκόσμια κοινότητα χρηστών, έτσι αν δεν βρούμε κάποιο εξάρτημα αρκεί μια απλή αναζήτηση στο google για να το βρούμε, να το κατεβάσετε και να το εισάγεται σε αυτό .

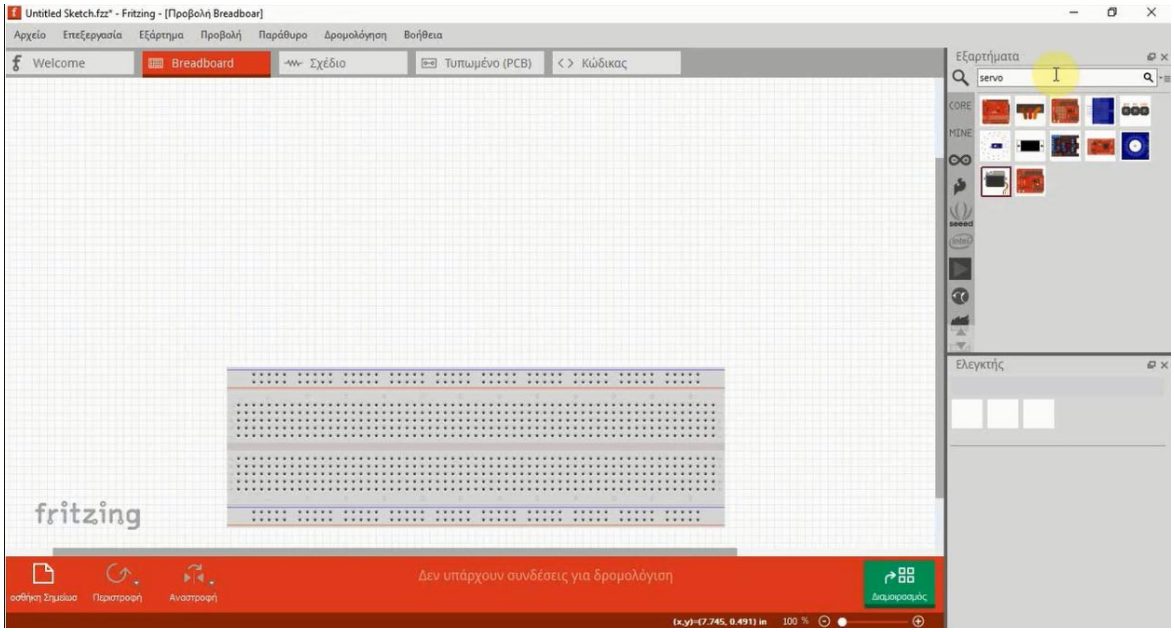
Εικόνα 6.1 Λογότυπο Fritzing

Η διαδικασία εγκατάστασης του Fritzing είναι πολύ απλή. Από τον ιστότοπο **http://fritzing.org/download/** κατεβάζουμε το πρόγραμμα και ακολουθούμε τις οδηγίες που περιγράφονται στον ιστότοπο. Μόλις ολοκληρωθεί η εγκατάσταση του ανοίγουμε τον πρόγραμμα από το εικονίδιο που δημιουργήθηκε και η οθόνη που θα βρεθούμε είναι η εξής.



Εικόνα 6.2 Αρχική σελίδα Fritzing

Στην επιλογή NewSketchκάνουμε κλικ για να δημιουργήσουμε ένα καινούριο Sketch ή εάν έχουμε ένα ήδη έτοιμο κάνουμε κλικ στην επιλογή OpenSketch για να το αναζητήσουμε. Εμείς θα δημιουργήσουμε ένα καινούριο Sketch για αυτό επιλέγουμε την πρώτη επιλογή. Η οθόνη που θα εμφανιστεί με το πάτημα του NewSketch θα είναι η εξής:



Εικόνα 6.3 Περιβάλλον δημιουργίας νέου sketchστο Fritzing

Η οθόνη που μας ανοίγει περιέχει ένα γραφικό περιβάλλον με τίποτα άλλο εκτός από ένα breadboard.Στα δεξιά της οθόνης μας βρίσκεται ένα searchbar στο οποίο θα αναζητήσουμε τα εξαρτήματα μας μέσα από τις δεκάδες βιβλιοθήκες εξαρτημάτων που περιέχει το Fritzing και ανανεώνονται αυτόματα σχεδόν καθημερινά.

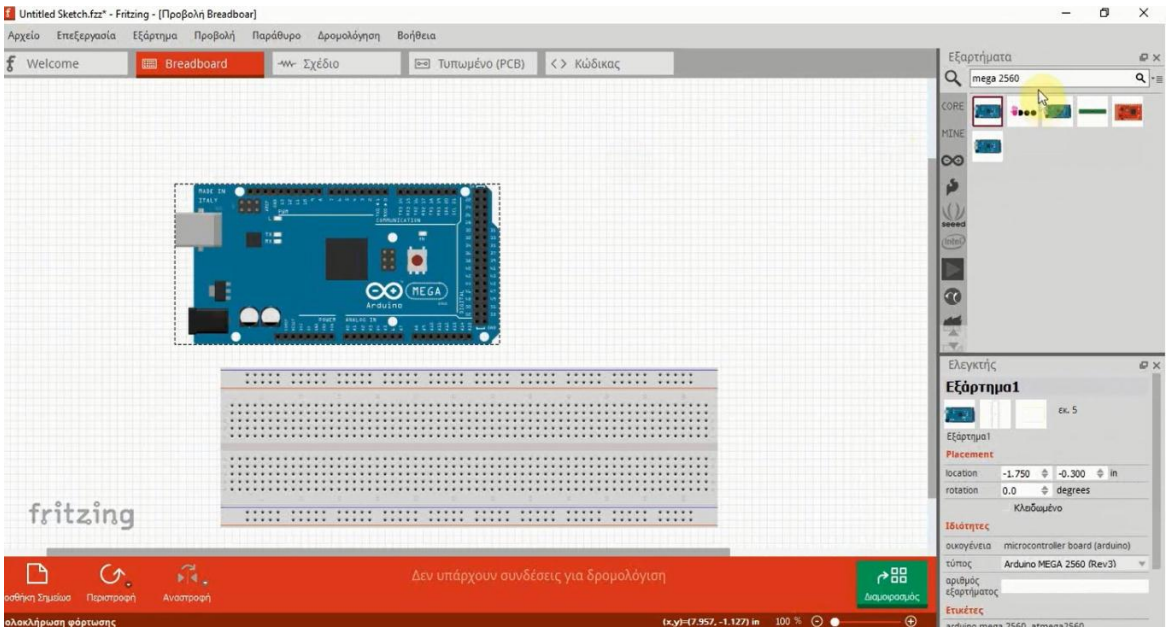
Ακόμη το Fritzing περιέχει προεγκατεστημένα εικονίδια από τις βιβλιοθήκες διάσημων κατασκευαστών ηλεκτρονικών εξαρτημάτων στα οποία εάν κάνουμε κλικ θα εμφανιστούν όλα τα εξαρτήματα του παρόντος κατασκευαστή. Κάτω από το searchbar υπάρχει ένα παράθυρο που ονομάζεται ελεγκτής το οποίο μας αναγράφει τις ιδιότητες και κάποιες βασικές πληροφορίες για το στοιχείο το οποίο έχουμε επιλέξει.

Κάτω από το γραφικό περιβάλλον υπάρχουν εργαλεία για εισαγωγή ενός άλλου σχεδίου μέσα στο υπάρχον Sketch για την περιστροφή και την αναστροφή εξαρτημάτων και για τις ρυθμίσεις προβολής του γραφικού περιβάλλοντος.

Το Fritzing παρέχει πολλές ακόμα λειτουργίες όπως η δημιουργία τυπωμένων πλακετών PCB διάφορων ηλεκτρονικών σχεδίων και η εισαγωγή κώδικα για προσομοίωση στο κύκλωμα μας αλλά δεν θα τις χρησιμοποιήσουμε στο συγκεκριμένο κύκλωμα που θα δημιουργήσουμε.

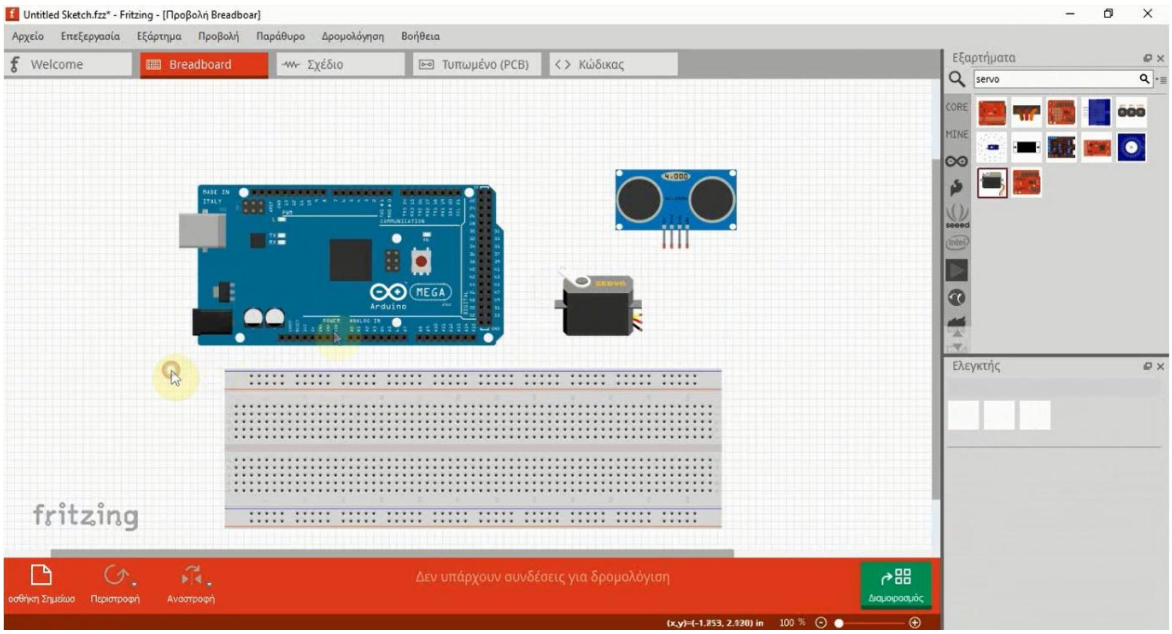
Ας ξεκινήσουμε λοιπόν να βρίσκουμε τα εξαρτήματα μας. Καθώς βρισκόμαστε στο Searchbar που αναφέραμε νωρίτερα , ο τρόπος εύρεσης των ηλεκτρονικών στοιχείων που χρειαζόμαστε είναι απλός. Εισάγουμε το όνομα του στοιχείου που θέλουμε και εάν υπάρχει το συγκεκριμένο στοιχείο εμφανίζεται στο κουτί κάτω από το searchbar.

Το πρώτο στοιχείο που θα αναζητήσουμε είναι το ArduinoMega 2560 που θα χρησιμοποιήσουμε στο πραγματικό μας κύκλωμα. Για συντομία πληκτρολογούμε **mega2560** στο searchbar και επιλέγουμε το σωστό στοιχείο από αυτά που μας εμφανίζει. Τώρα μας μένει να τοποθετήσουμε το Arduino μας στο γραφικό περιβάλλον που υπάρχει δίπλα και βρίσκεται το breadboard. Αυτό γίνεται με την λειτουργία drag&drop που μας παρέχει το Fritzing , έτσι απλά επιλέγουμε και σέρνουμε το Arduino μας δίπλα και πάνω από το breadboard. Το τελικό αποτέλεσμα μόλις σύρουμε το Arduino δίπλα θα είναι το εξής:



Εικόνα 6.4 Εισαγωγή ArduinoMega 2560 στο sketch

Με τον ακόλουθο τρόπο θα τοποθετήσουμε τον αισθητήρα υπέρυθρωνHC-SR04 και το Servo μοτεράκι μας γράφοντας **hc-sr04** και **servo** αντίστοιχα στο searchbar. Τα σέρνουμε δίπλα και το αποτέλεσμα είναι το εξής:

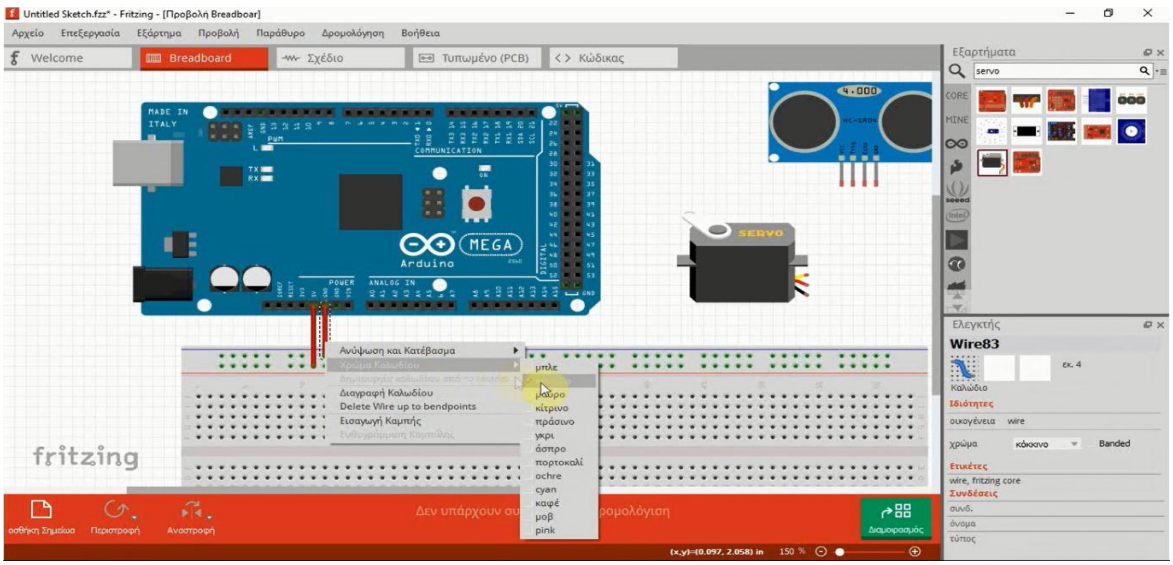


Εικόνα 6.5 Εισαγωγή HC-SR04 και σερβοκινητήραστο sketch

6.1 Σύνδεση εξαρτημάτων με Arduino

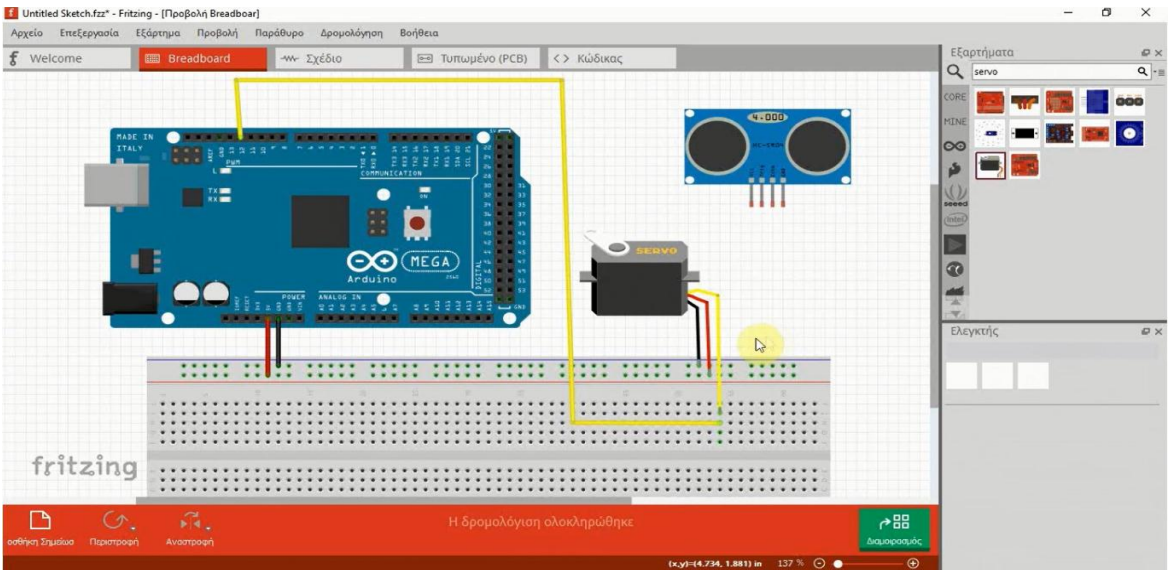
Το επόμενο μας βήμα είναι η σύνδεση των εξαρτημάτων μεταξύ τους μέσω του breadboard. Το πρώτο μας βήμα θα είναι να συνδέσουμε τα pin**5V** και **GND** (γείωση) του Arduino με το breadboard. Με την ίδια λειτουργία drag&drop που βγάλαμε τα εξαρτήματα από το searchbar κάνουμε κλικ στο pi που θέλουμε και σέρνουμε το καλώδιο που δημιουργείται μέχρι το επιθυμητό pin του breadboard.

Ακόμη για την ευκολία αναγνώρισης των καλωδίων εάν κάνουμε δεξί κλικ πάνω σε ένα καλώδιο από την επιλογή **Χρώμα Καλωδίου** αλλάζουμε το χρώμα σε όποιο άλλο θέλουμε.



Εικόνα 6.6 Αλλαγή χρώματος καλωδίων

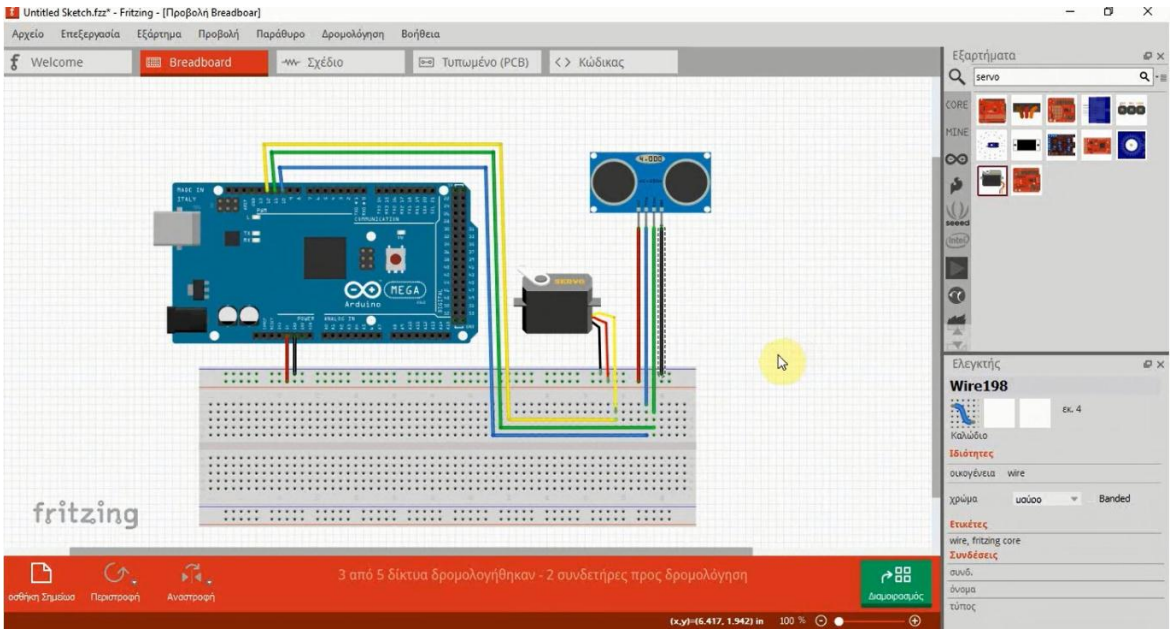
Μετά συνδέουμε το Servomotor μας με τα pin 5V και την γείωση του breadboard τα οποία θα τροφοδοτούνται μέσω του Arduino μας. Τον παλμό του Servo θα τον συνδέσουμε με το **pin12** του Arduino μας μέσω του breadboard και θα το ρυθμίσουμε στο αντίστοιχο κίτρινο χρώμα του.



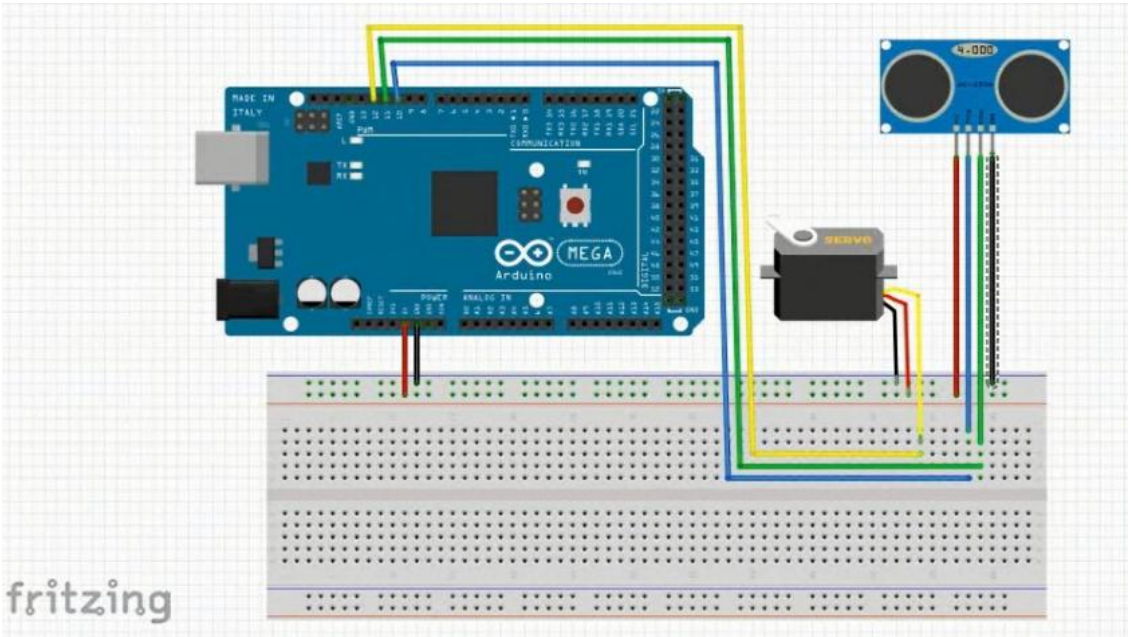
Εικόνα 6.7 ΣύνδεσηServo pulse me pin12

Το **pin12** θα μπορούσε να είναι οποιοδήποτε άλλο digitalpin αλλά επιλέξαμε αυτό για τον κώδικα μας . Το επόμενο μας βήμα θα είναι συνδέσουμε τον αισθητήρα HC-SR04 . Το πρώτο pin του αισθητήρα το **Vcc** το συνδέουμε με την γραμμή 5V που τροφοδοτεί το breadboard μας, Το pin**Trig** (trigger) με το **pin10** του Arduino μέσω του breadboard και το pin**Echo** με το **pin11** του Arduino.

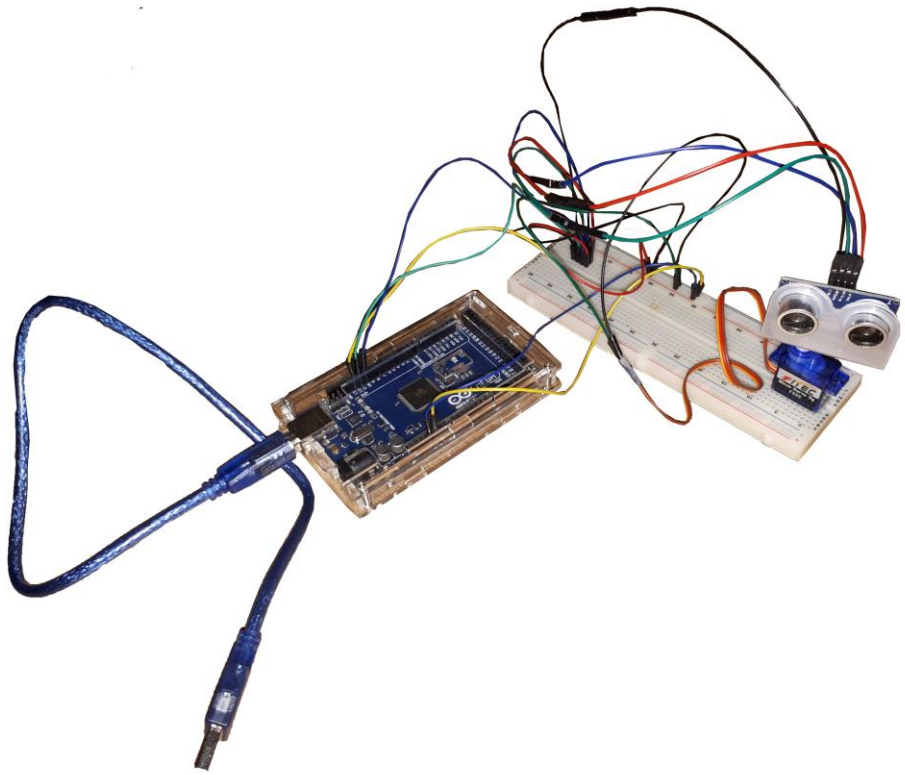
Θα μπορούσαμε να χρησιμοποιήσουμε και εδώ διαφορετικά pin απλά θα έπρεπε να τα δηλώσουμε μέσα στον κώδικα μας. Το τελευταίο pin του αισθητήρα **Gnd** θα το συνδέσουμε στην γραμμή της γείωση που έχουμε συνδέσει στο breadboard μέσω του Arduino. Αλλάζουμε τα χρώματα τον καλωδίων για να είναι πιο εύκολη η προσέγγιση των λειτουργιών του κάθε καλωδίου με το συνδεδεμένο pin και το αποτέλεσμα μας είναι το εξής.



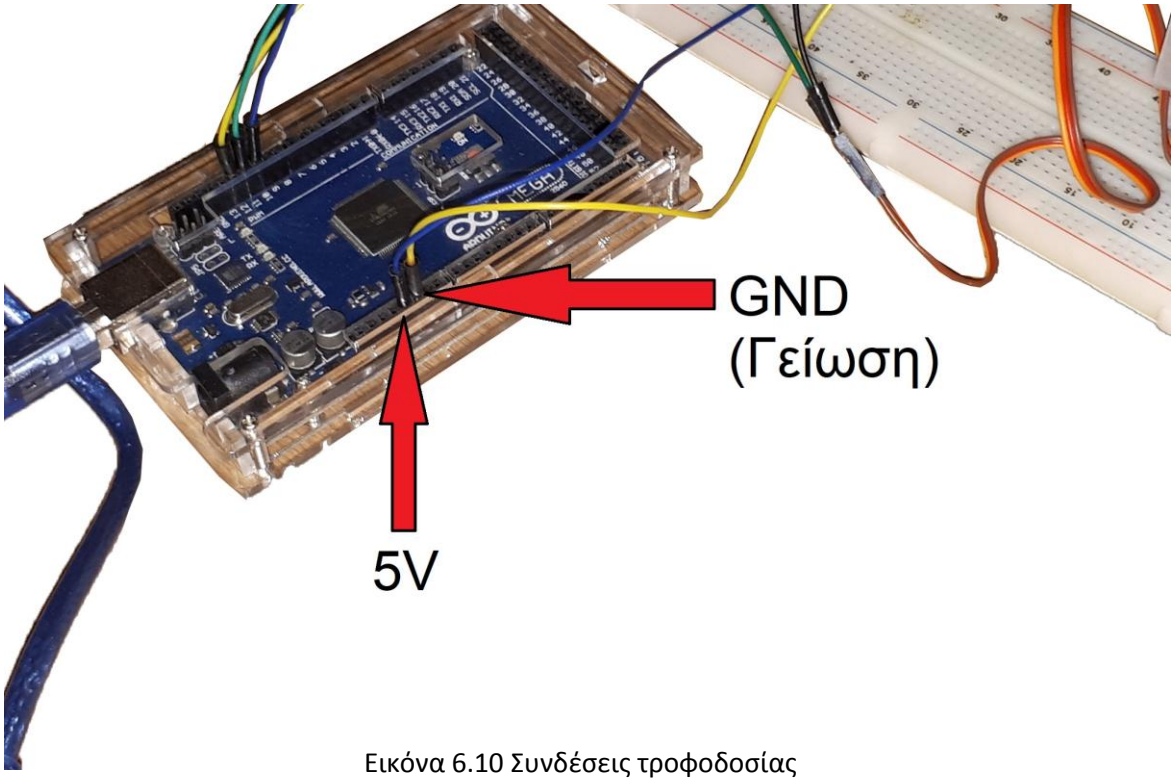
Έχοντας αυτήν την καθαρή είκόνα του κυκλώματος μας θα είναι πολύ ευκολότερο να δημιουργήσουμε το πραγματικό μας κύκλωμα



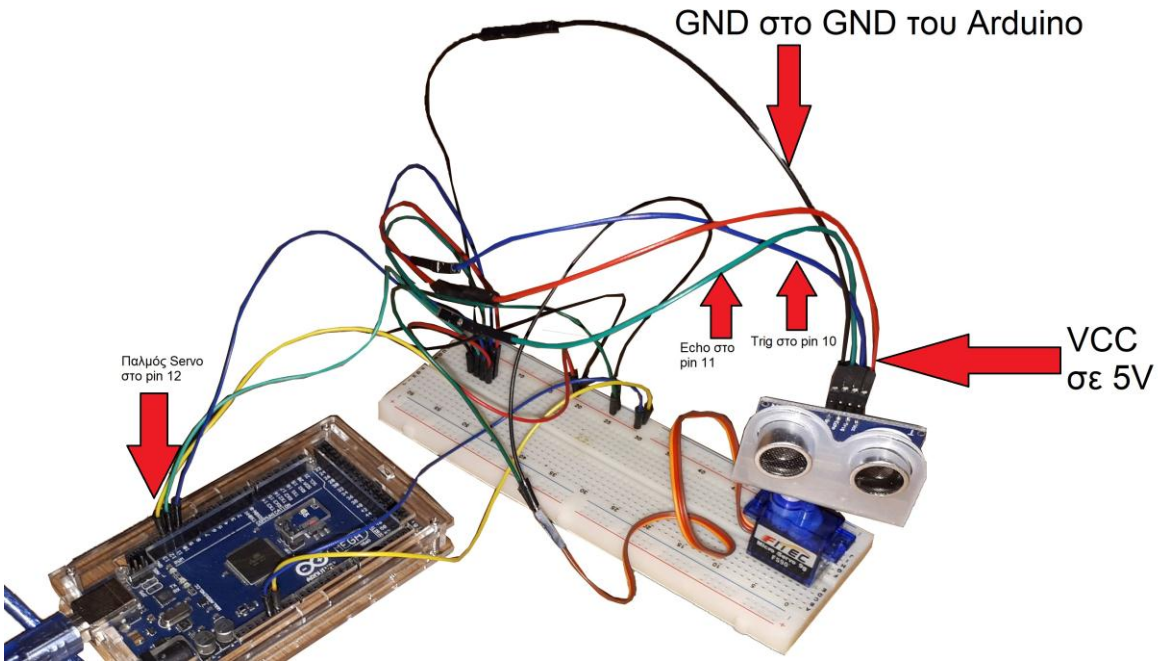
Μετά από λίγη ώρα συναρμολόγησης, το κύκλωμα μας έχειως εξής:



Εικόνα 6.9Ολοκλήρωση συνδέσεωνφυσικού κυκλώματος



Εικόνα 6.10 Συνδέσεις τροφοδοσίας



Εικόνα 6.11 Συνδέσεις σερβοκινητήρα και αισθητήρα HC-SR04

Με τα υλικά στήριξης που υπήρχαν μέσα στην συσκευασία του Servomotor που αγοράσαμε τοποθετήσαμε πάνω στον μηχανισμό μια βάση και τον αισθητήρα υπέρυθρων HC-SR04 για να μιμηθεί την περιστροφική λειτουργία των Sonar μέσω την κίνησης που θα προγραμματίσουμε να κάνει το Servomotor.

Θα πρέπει να σιγουρευτούμε ότι όλα τα καλώδια κάνουν καλή επαφή μεταξύ τους και με τα pins και πως το servomotor έχει τοποθετηθεί σωστά πάνω στο breadboard έτσι ώστε να μην πέσει κατά την διάρκεια της κίνησης του σερβομηχανισμού.

7. Προγραμματισμός Arduino μέσω Arduino IDE

Αφότου ολοκληρώθηκε η δημιουργία του φυσικού μας κυκλώματος , το επόμενο μας βήμα είναι να δημιουργήσουμε το πρόγραμμα με το οποίο θα συγχρονιστεί ο αισθητήρας HC-SR04 έτσι ώστε να μας παρέχει τις σωστές μετρήσεις απόστασης , ο σερβοκινητήρας μας να εκτελεί την σωστή περιστροφή, στον σωστό χρόνο για να λειτουργεί το κύκλωμα μας όπως ένα σύστημα Sonar και να ανιχνεύει ότι εμπόδιο βρίσκεται στο βεληνεκές του. Θα πρέπει να δηλώσουμε ακόμη τα σωστά pin για την κάθε λειτουργία του κάθε συνδεμένου με το Arduino μας εξαρτήματος , να προσθέσουμε τις βιβλιοθήκες που χρειάζονται για την σωστή λειτουργία των εξαρτημάτων και να δημιουργήσουμε όλες τις απαραίτητες λειτουργίες για το κύκλωμα μας.

Παρακάτω παρατήθεται ο κώδικας του προγράμματος που δημιουργήθηκε για την λειτουργία του κυκλώματος μας και θα αναλυθεί η κάθε γραμμή του κώδικα . Περιληπτικά το πρόγραμμα ξεκίνησε με την προσθήκη της βιβλιοθήκης **Servo.h** η μέσα από την οποία θα ορίσουμε τις λειτουργίες του σερβοκινητήρα μας. Ορίσαμε τα pinTrig και Echo του αισθητήρα HC-SR04 τα οποία εκπέμπουν και λαμβάνουν τα υπερηχτικά κύμματα όπως ακριβώς και τα συστήματα τύπου Sonar.

Δηλώσαμε τις μεταβλητές για την διάρκει και την απόσταση , **duration** και **distance** αντίστοιχα και μετά δημιουργήσαμε ένα Servo αντικείμενο , το οποίο ονομάσαμε **myServo** για τον χειρισμό του σερβοκινητήρα μας.

Το επόμενο μας βήμα είναι να καλέσουμε την **setup()** του προγράμματος μέσα στην οποία θα ορίσουμε το pin**Trig** του αισθητηρα υπερύθρων ώς έξοδο (**Output**) και το pin**Echo** ως είσοδο (**Input**). Ρυθμίσαμε το baudrate στα **9600**bps για την σειριακή μετάδοση δεδομένωνκαι ορίσαμε το **pin 12** στο κώδικα μας ώς το pin στο οποίο έχουμε τοποθετήσει τον παλμό του σερβοκινητήρα μας.

Μετά το κάλεσμα της setup() ήρθε η στιγμή να καλέσουμε την **loop()** η οποία θα επαναλαμβάνει τις λειτουργίες τις οποίες έχουμε ρυθμίσει μέσα της. Θα περιστρέφει τον σερβοκινητήρα από τις 15 έως τις 165 μοίρες , και θα γράφει την τιμή της κάθε γωνίας με καθυστέρηση 30 millisecond και θα καλεί μια λειτουργία για τον υπολογισμό της απόστασης που μετριέται από τον αισθητήρα υπέρυθρων για κάθε μοίρα.

Στην συνέχεια θα στέλνει την τιμή της μοίρας που βρίσκεται ο σερβομηχανισμός του σερβοκινητήρα μας στη σειριακή θύρα (SerialPort), ένα χαρακτήρα " **,** " ο οποίος θα μας χρησιμεύσει αργότερα στο ProcessingIDE , θα στέλνει την τιμή της απόστασης στην σειριακή θύρα και ένα χαρακτήρα " **.** " για όπου και αυτός θα χρησιμεύσει αργότερα για το ProcessingIDE.

Η ίδια ακριβώς λειτουργία θα επαναλαμβάνεται αυτή την φορά από τις 165 μοίρες έως στις 15 για να μιμείται την λειτουργία των Sonar.

Μετά το τέλος της loop() θα ορίζεται η λειτουργία **calculateDistance()** για τον υπολογισμό την μετρούμενης απόστασης από τον αισθητήρα υπέρυθρων.

Topin**Trig** θα παραμένει σε κατάσταση **Low** για 2 microseconds και μετά θα το θέτουμε σε κατάσταση **High** και θα παραμένει για 10 microseconds ώσπου να ξαναγυρίσει σε **Low**. Η μεταβλητή **duration** θα διαβάσει το pin**Echo** του HC-SR04 και θα επιστρέφει τον χρόνο της απόστασης που ταξίδεψε ο ήχος σε microseconds.

Η μεταβλητή της απόστασης θα ισούται με την μεταβλητή της διάρκειας

$$\frac{duration\ X\ 0.034}{2}$$

Και στο τέλος θα μας επιστρέφεται η απόσταση distance.

7.1 Κώδικας στο Arduino IDE

Sonar | Arduino 1.8.3

Αρχείο Επεξεργασία Σχέδιο Εργαλεία Βοήθεια

Sonar §

```
#include <Servo.h>.

const int trigPin = 10;
const int echoPin = 11;
long duration;
int distance;

Servo myServo;

void setup() {
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  Serial.begin(9600);
  myServo.attach(12);
}
void loop() {
  for (int i = 15; i <= 165; i++) {
    myServo.write(i);
    delay(30);
    distance = calculateDistance();

    Serial.print(i);
    Serial.print(",");
    Serial.print(distance);
    Serial.print(".");
  }
  for (int i = 165; i > 15; i--) {
    myServo.write(i);

    delay(30);
    distance = calculateDistance();
    Serial.print(i);
    Serial.print(",");
    Serial.print(distance);
    Serial.print(".");
  }
}
int calculateDistance() {

  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  duration = pulseIn(echoPin, HIGH);
  distance = duration * 0.034 / 2;
  return distance;
}
```

Εικόνα7.1 Κώδικας στο Arduino IDE

#include <Servo.h>.	Προσθέτουμε την βιβλιοθήκη Servo.h έτσι ώστε να μπορούμε να δηλώσουμε την λειτουργία που θέλουμε να κάνει ο σερβοκινητήρας μας μέσω του κώδικα μας.
const int trigPin = 10;	Δηλώνουμε το pinTrig του αισθητήρα υπέρυθρων που θα εκπέμπει το υπερηχητικό κύμα
const int echoPin = 11;	Δηλώνουμε το pinEcho του αισθητήρα υπέρυθρων που θα δέχεται το υπερηχητικό κύμα
long duration;	Δηλώνουμε την μεταβλητή της διάρκειας (duration)
int distance;	Δηλώνουμε την μεταβλητή της απόστασης (distance)
Servo myServo;	Δημιουργούμε ένα αντικείμενο servo (servoobject) για να μπορούμε να ελέγξουμε τον σερβοκινητήρα μας
void setup() {	Καλούμε την setupη οποία καλείτε όταν ξεκινάει το sketch και μέσα δηλώνουμε τις μεταβλητές μας, τα pin κλπ.
pinMode(trigPin, OUTPUT);	Ορίζει το pinTrig του αισθητήρα μας ως έξοδο (Output)
pinMode(echoPin, INPUT);	Ορίζει το pinTrig του αισθητήρα μας ως είσοδο (Input)
Serial.begin(9600);	Ρύθμιση του ρυθμού baud (baudrate) για σειριακή μετάδοση δεδομένων

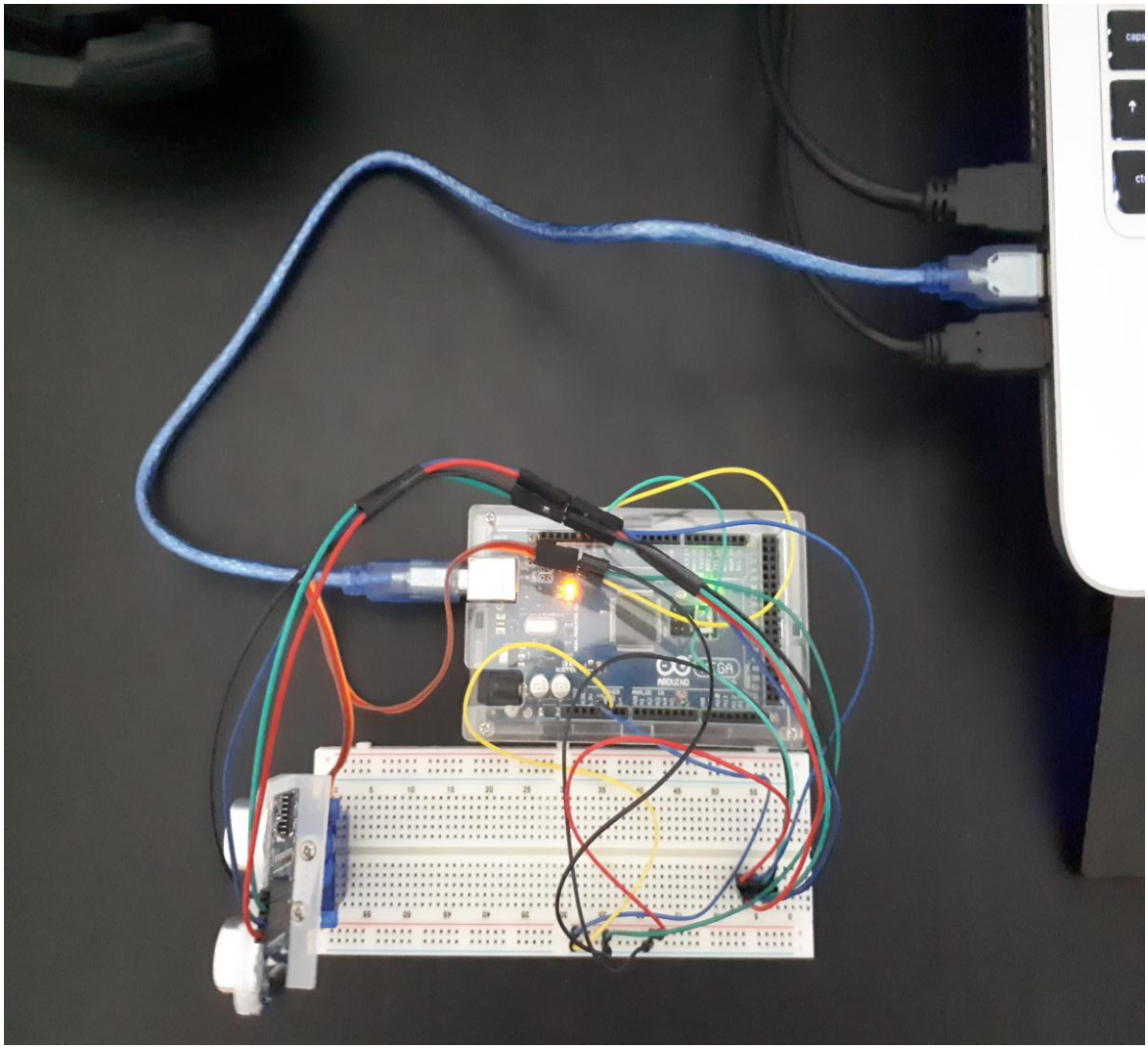
myServo.attach(12); // Defines on which pin is the servo motor attached
}
	Προσδιορίζει σε pin του Arduino μας έχει συνδεθεί ο παλμός (pulse) του σερβοκινητήρα. Στην περίπτωση μας το pin 12 όπως αναφέραμε και νωρίτερα.

void loop() {	Μετά από το κάλεσμα της setupλειτουργίας η loop κάνει ακριβώς ότι λέει το όνομα της, επαναλαμβάνει τις λειτουργίες που τις έχουμε ορίσει.
for(int i=15;i<=165;i++){	Περιστροφή του σερβοκινητήρα από 15 έως 165 μοίρες
myServo.write(i);	Γράφει μία τιμή στο servo, δηλαδή ορίζει την γωνία του οδηγού του σερβοκινητήρα
delay(30);	Καθυστέρηση αλλαγής της γωνίας 30 milliseconds
distance = calculateDistance();	Καλεί μια λειτουργία για τον υπολογισμό της απόστασης που μετριέται από τον αισθητήρα υπέρυθρων για κάθε μοίρα
Serial.print(i);	Στέλνει την τιμή της μοίρας που βρίσκεται ο σερβομηχανισμός του σερβοκινητήρα μας στη σειριακή θύρα (SerialPort)
Serial.print(",");	Στέλνει ένα επιπλέον χαρακτήρα (,) δίπλα στη προηγούμενη τιμή , ο οποίο χρειάζεται αργότερα στο ProcessingIDE
Serial.print(distance);	Στέλνει την τιμή της απόστασης στην σειριακή θύρα (SerialPort)
Serial.print("."); **}**	Στέλνει ένα επιπλέον χαρακτήρα (.) δίπλα στη προηγούμενη τιμή , ο οποίο χρειάζεται αργότερα στο ProcessingIDE
for(int i=165;i>15;i--){	Επαναλαμβάνει τις προηγούμενες γραμμές από τις 165 μοίρες έως τις 15.
myServo.write(i);	Γράφει μία τιμή στο servo, δηλαδή ορίζει την γωνία του οδηγού του σερβοκινητήρα όπως προηγουμένως
delay(30);	Καθυστέρηση αλλαγής της γωνίας 30 milliseconds όπως προηγουμένως
distance = calculateDistance();	Καλεί την λειτουργία για την μέτρηση της απόστασης για κάθε μοίρα

Serial.print(i);	Στέλνει την τιμή της μοίρας που βρίσκεται ο σερβομηχανισμός του σερβοκινητήρα μας στη σειριακή θύρα (SerialPort)
Serial.print(",");	Στέλνει ένα επιπλέον χαρακτήρα (,) δίπλα στη προηγούμενη τιμή , ο οποίο χρειάζεται αργότερα στο ProcessingIDE
Serial.print(distance);	Στέλνει την τιμή της απόστασης στην σειριακή θύρα (SerialPort)
Serial.print("."); **}** **}**	Στέλνει ένα επιπλέον χαρακτήρα (.) δίπλα στη προηγούμενη τιμή , ο οποίο χρειάζεται αργότερα στο ProcessingIDE και τελειώνει η loop
int calculateDistance(){	Ορισμός της λειτουργίας για τον υπολογισμό την μετρούμενης απόστασης από τον αισθητήρα υπέρυθρων
digitalWrite(trigPin, LOW);	Θέτει το pinTrig του αισθητήρα σε κατάσταση Low
delayMicroseconds(2);	Η λειτουργία σε κατάσταση Low παραμένει για 2 microsecond
digitalWrite(trigPin, HIGH);	Θέτει το pinTrig του αισθητήρα σε κατάσταση High
delayMicroseconds(10);	Η λειτουργία σε κατάσταση High παραμένει για 10 microsecond
digitalWrite(trigPin, LOW);	Θέτει το pinTrig του αισθητήρα σε κατάσταση Low
duration = pulseIn(echoPin, HIGH);	Διαβάζει την μέτρηση από το pinEcho του HC-SR04 και επιστρέφει τον χρόνο μετάδοσης του ηχητικού κύματος σε microseconds
distance= duration*0.034/2;	Η αριθμητική πράξη για την σωστή μέτρηση της απόστασης .
return distance; **}**	Επιστροφή της τιμής της απόστασης και τέλος του κώδικά μας

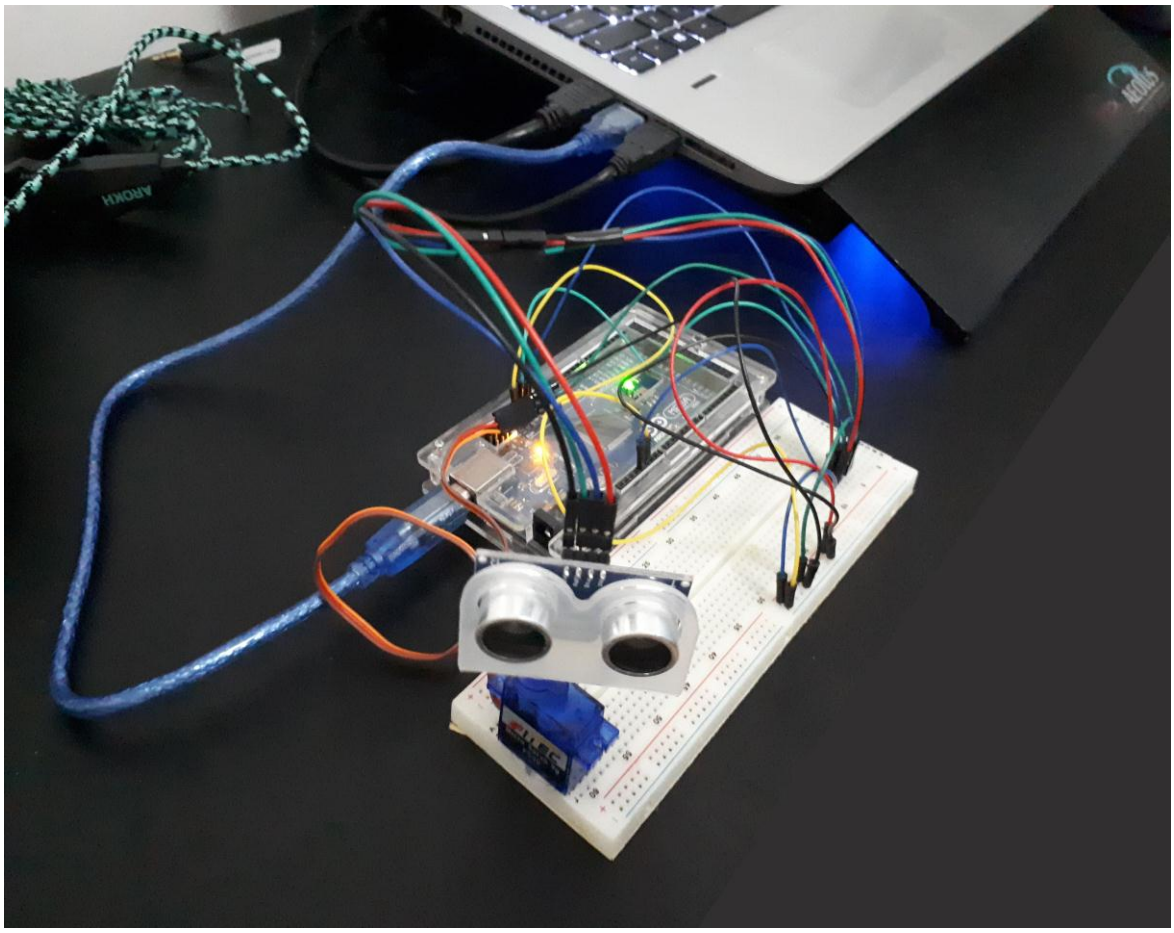
Φόρτωση κώδικα στο Arduino

Το επόμενο μας βήμα είναι να ανεβάσουμε το πρόγραμμα το οποίο χρησιμοποιήσαμε στο ArduinoMega 2560 μας μέσω της θύρα usb του υπολογιστή μας. Συνδέουμε το καλώδιο usb που βρίσκεται στην συσκευασία του Arduino με τον μικροελεγκτή και με μια θύρα usb του υπολογιστή μας.



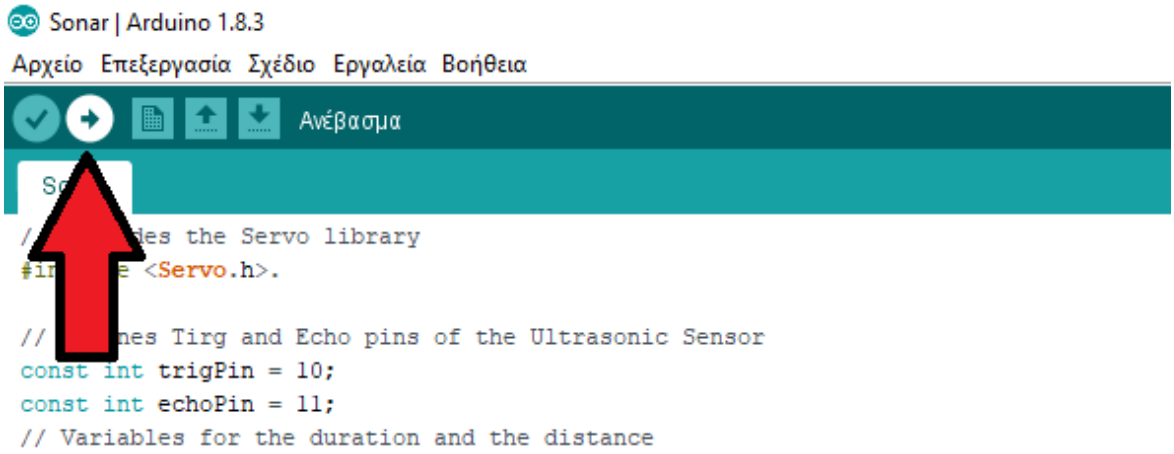
Εικόνα7.2 Σύνδεση του Arduinoστο ρcγια ανέβασμα του κώδικα

Το On led του Arduino ανάβει πράσινο, δείχνοντας μας ότι η συσκευή είναι ενεργοποιημένη και είναι έτοιμη για να φορτώσουμε το πρόγραμμα μας στην μνήμη της. Ο σερβοκινητήρας έχει τοποθετηθεί πάνω στο breadboard με σιλικόνη για να μην υπάρχει πρόβλημα με την στήριξη καθώς ο σερβομηχανισμός περιστρέφεται και ο αισθητήρας HC-SR04 βρίσκεται στα 6.3 εκατοστά από το έδαφος.



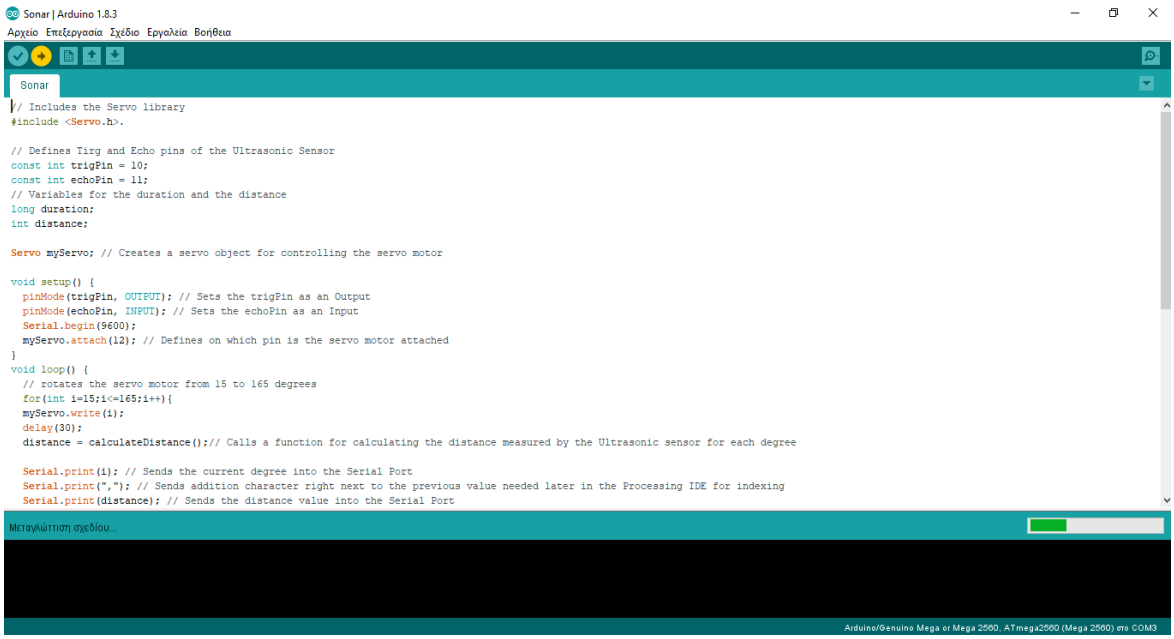
Εικόνα7.3Σύνδεση του Arduinoστο pcγια ανέβασμα του κώδικα

Επόμενο βήμα είναι να πατήσουμε το κουμπί για το ανέβασμα (upload) του κώδικα στον μικροελεγκτή στο ArduinoIDE.



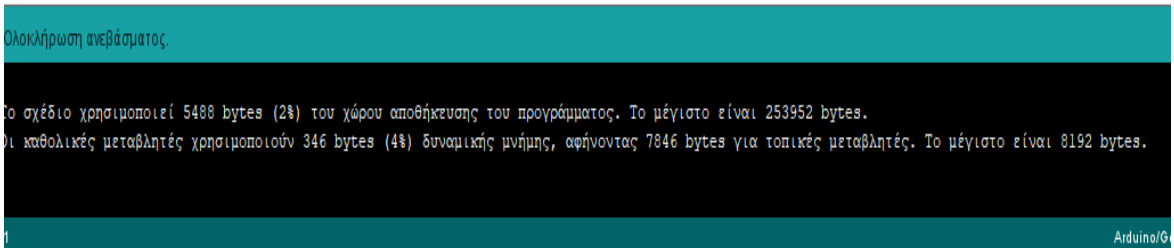
Εικόνα7.4Κουμπί λειτουργίας στο ArduinoIDEγια ανέβασμα του κώδικα

Το κουμπί γίνεται κίτρινο και το ProgressBar πάνω από την περιοχή μηνυμάτων φορτώνει σιγά – σιγά.

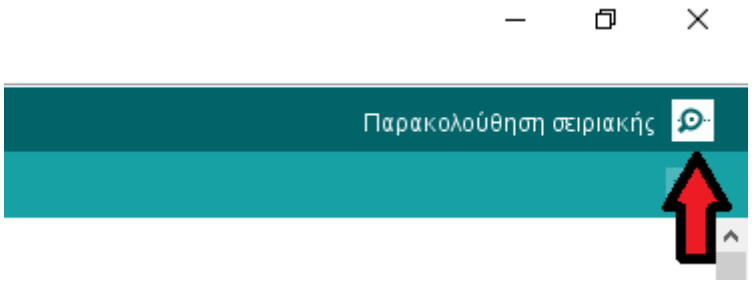


Εικόνα7.5Διαδικασία ανεβάσματος του κώδικα

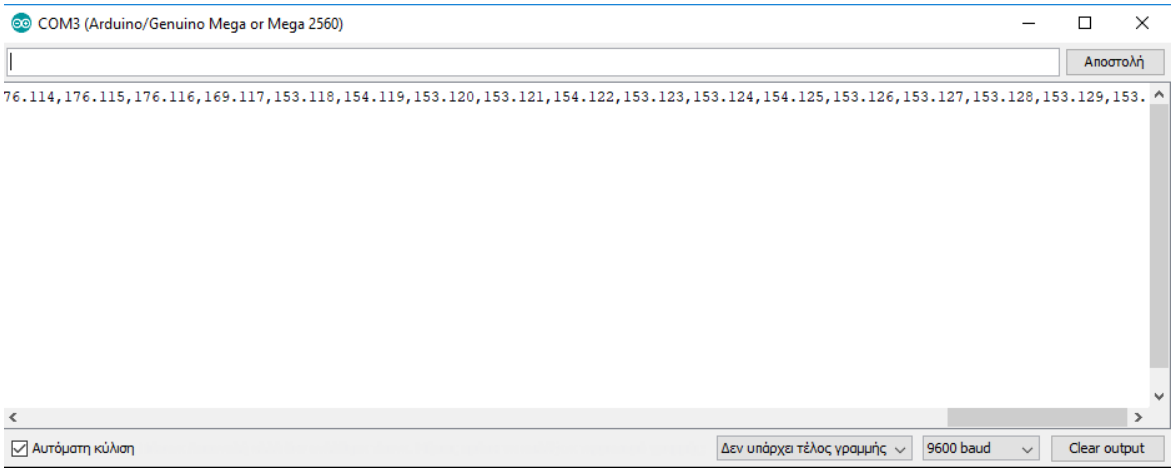
Η φόρτωση του κώδικα ολοκληρώνεται και στην περιοχή των μηνυμάτων μας προβάλλεται το μήνυμα “**Ολοκλήρωση ανεβάσματος**” και ο χώρος που καταλαμβάνει το πρόγραμμα σε bytes. Το πρόγραμμα μας χρησιμοποιεί 5488 bytes ενώ το μέγιστο που θα μπορούσε να χρησιμοποιήσει είναι 253952 bytes.
Οι καθολικές μεταβλητές το 4% της δυναμικής μνήμης (346 bytes) από το μέγιστο που είναι 8192 bytes.



Ο σερβοκινητήρας μας αρχίζει να περιστρέφεται άρα το κύκλωμα μας λειτουργεί και αν ανοίξουμε την σειριακό monitor στο ArduinoIDE από το κουμπί



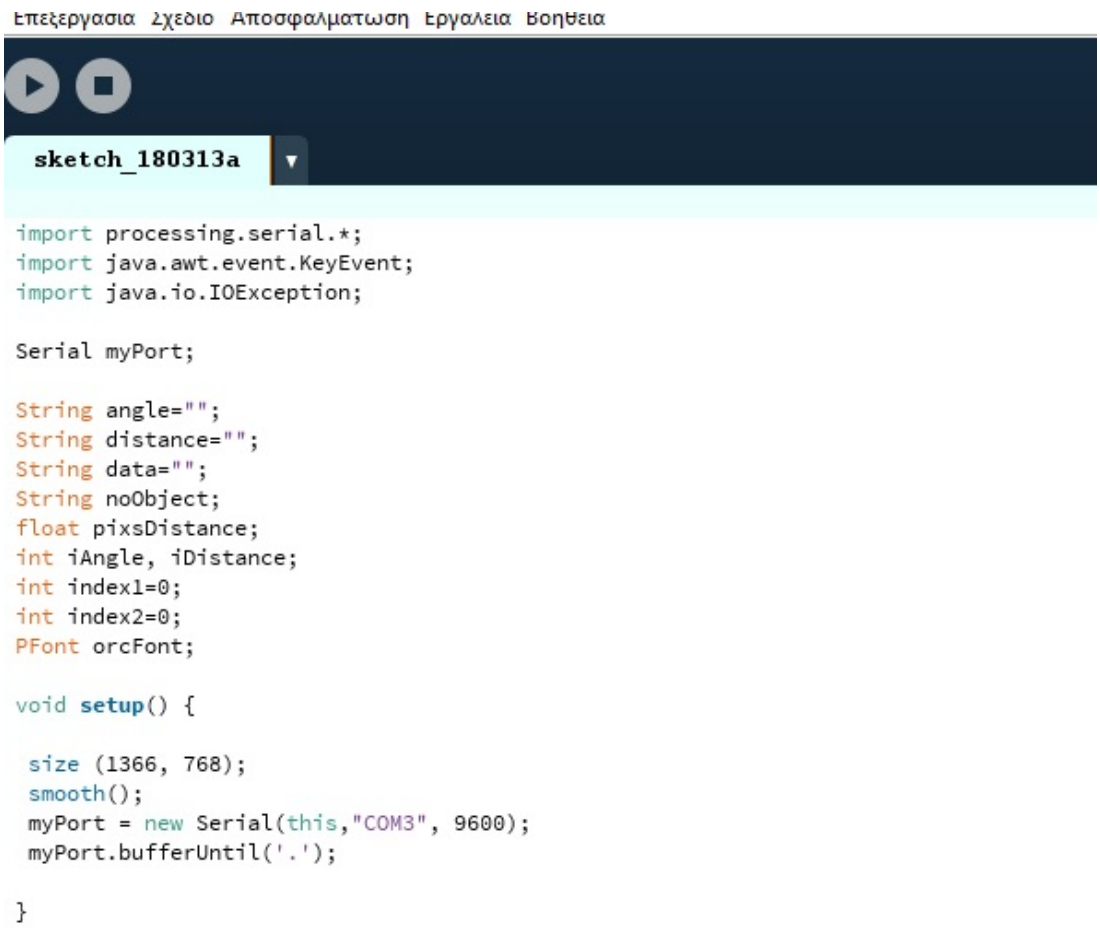
Θα δούμε ότι στην οθόνη προβάλλονται διάφορες τιμές χωρίς να μπορούμε να καταλάβουμε τι ακριβώς αντιπροσωπεύουν.



Εικόνα7.6Παρακολούθηση σειριακής κατά την λειτουργία του κυκλώματος

Το Επόμενο στάδιο της εργασίας μας είναι η δημιουργία του γραφικού περιβάλλοντος απεικόνισης του Sonar μας μέσω του ProcessingIDE.

8. Κώδικας γραφικού περιβάλλοντος στο ProcessingIDE




```
import processing.serial.*;
import java.awt.event.KeyEvent;
import java.io.IOException;

Serial myPort;

String angle="";
String distance="";
String data="";
String noObject;
float pixsDistance;
int iAngle, iDistance;
int index1=0;
int index2=0;
PFont orcFont;

void setup() {

 size (1366, 768);
 smooth();
 myPort = new Serial(this,"COM3", 9600);
 myPort.bufferUntil('.');

}
```

```
void draw() {

  fill(98,245,31);
  noStroke();
  fill(0,4);
  rect(0, 0, width, height-height*0.065);

  fill(98,245,31);

  drawRadar();
  drawLine();
  drawObject();
  drawText();
}

void serialEvent (Serial myPort) {

  data = myPort.readStringUntil('.');
  data = data.substring(0,data.length()-1);

  index1 = data.indexOf(",");
  angle= data.substring(0, index1);
  distance= data.substring(index1+1, data.length());


  iAngle = int(angle);
  iDistance = int(distance);
}

void drawRadar() {
  pushMatrix();
  translate(width/2,height-height*0.074);
  noFill();
  strokeWeight(2);
  stroke(98,245,31);

  arc(0,0,(width-width*0.0625),(width-width*0.0625),PI,TWO_PI);
  arc(0,0,(width-width*0.27),(width-width*0.27),PI,TWO_PI);
  arc(0,0,(width-width*0.479),(width-width*0.479),PI,TWO_PI);
  arc(0,0,(width-width*0.687),(width-width*0.687),PI,TWO_PI);
  // draws the angle lines
  line(-width/2,0,width/2,0);
  line(0,0,(-width/2)*cos(radians(30)),(-width/2)*sin(radians(30)));
  line(0,0,(-width/2)*cos(radians(60)),(-width/2)*sin(radians(60)));
  line(0,0,(-width/2)*cos(radians(90)),(-width/2)*sin(radians(90)));
  line(0,0,(-width/2)*cos(radians(120)),(-width/2)*sin(radians(120)));
  line(0,0,(-width/2)*cos(radians(150)),(-width/2)*sin(radians(150)));
  line((-width/2)*cos(radians(30)),0,width/2,0);
  popMatrix();
}

void drawObject() {
  pushMatrix();
  translate(width/2,height-height*0.074);
  strokeWeight(9);
  stroke(255,69,0);
  pixsDistance = iDistance*((height-height*0.1666)*0.025);

  if(iDistance<40){


 line(pixsDistance*cos(radians(iAngle)),-pixsDistance*sin(radians(iAngle)),(width-width*0.505)*cos(radians(iAngle)),-(width-width*0.505)*sin(radians(iAngle)));
 }
 popMatrix();
}
```

```
void drawLine() {
  pushMatrix();
  strokeWeight(9);
  stroke(30,250,60);
  translate(width/2,height-height*0.074);
  line(0,0,(height-height*0.12)*cos(radians(iAngle)),-(height-height*0.12)*sin(radians(iAngle)));
  popMatrix();
}

void drawText() {

  pushMatrix();
  if(iDistance>40) {
  noObject = "Out of Range";
  }
  else {
  noObject = "In Range";
  }
  fill(0,0,0);
  noStroke();
  rect(0, height-height*0.0648, width, height);
  fill(98,245,31);
  textSize(25);
  text("10cm",width-width*0.3854,height-height*0.0833);
  text("20cm",width-width*0.281,height-height*0.0833);
  text("30cm",width-width*0.177,height-height*0.0833);
  text("40cm",width-width*0.0729,height-height*0.0833);
  textSize(40);
  text("Object: " + noObject, width-width*0.875, height-height*0.0277);
  text("Angle: " + iAngle +" °", width-width*0.48, height-height*0.0277);
  text("Distance: ", width-width*0.26, height-height*0.0277);
  if(iDistance<40) {
  text("        " + iDistance +" cm", width-width*0.225, height-height*0.0277);
  }
  textSize(25);
  fill(98,245,60);
  translate((width-width*0.4994)+width/2*cos(radians(30)),(height-height*0.0907)-width/2*sin(radians(30)));
  rotate(-radians(-60));
  text("30°",0,0);
  resetMatrix();
  translate((width-width*0.503)+width/2*cos(radians(60)),(height-height*0.0888)-width/2*sin(radians(60)));
  rotate(-radians(-30));
  text("60°",0,0);
  resetMatrix();
  translate((width-width*0.507)+width/2*cos(radians(90)),(height-height*0.0833)-width/2*sin(radians(90)));
  rotate(radians(0));
  text("90°",0,0);
  resetMatrix();
  translate(width-width*0.513+width/2*cos(radians(120)),(height-height*0.07129)-width/2*sin(radians(120)));
  rotate(radians(-30));
  text("120°",0,0);
  resetMatrix();
  translate((width-width*0.5104)+width/2*cos(radians(150)),(height-height*0.0574)-width/2*sin(radians(150)));
  rotate(radians(-60));
  text("150°",0,0);
  popMatrix();
}
```

Εικόνα8.1 Κώδικας στο Processing IDE

Αρχικά εισάγουμε την βιβλιοθήκη για σειριακή επικοινωνία και την βιβλιοθήκη για την ανάγνωση δεδομένων από την σειριακή θύρα. Ορίζουμε ένα Serial αντικείμενο (object) με την **Serial myPort;** και μετά δηλώνουμε τις μεταβλητές που χρειάζονται για την δημιουργία του γραφικού περιβάλλοντος όπως η μοίρες, η απόσταση, τα δεδομένα, οι γραμματοσειρές κλπ.

Μέσα στην **setup**() μας δηλώνουμε με την **size** την ανάλυση της οθόνης μας για την σωστή απεικόνιση της εφαρμογής, την σειριακή θύρα που είναι τοποθετημένοο το Arduino, το ρυθμό baud και μετά ξεκινάει η σειριακή επικοινωνία.
Με την **myPort.bufferUntil('.');** διαβάζονται τα δεδομένα από την σειριακή θύρα μέχρι τον χαρακτήρα " . " που δηλώσαμε νωρίτερα , άρα τα δεδομένα που διαβάζονται είναι τα **angle** και **distance** , γωνία και απόσταση αντίστοιχα.

Στην draw() θα δημιουργήσουμε το γραφικό μας περιβάλλον. Θα δηλώσουμε το επιθυμητό χρώμα που θέλουμε να εμφανίζονται οι άξονες , οι μοίρες , οι γραμμές ,το κείμενο και θα τα σχεδιάσουμε με τις **drawRadar**(); ,**drawLine**(); , **drawObject**(); , **drawText**(); . Το πράσινο χρώμα που επιλέξαμε στο RGB έχει τις τιμές που ορίζονται στο fill(**98**,**245**,**31**).

Εικόνα8.2 Χρώμα γραφικού περιβάλλοντος Sonar

Στην **serialEvent** η οποία αντλεί τα δεδομένα την από την σειριακή θύρα με την **data = myPort.readStringUntil('.');** διαβάζει τα δεδομένα μέχρι τον χαρακτήρα " . " και τα αποθηκεύει στην μεταβλητή **data**.

Η **index1 = data.indexOf(",");**βρίσκει τον χαρακτήρα “ , “ και τον τοποθετεί στην μεταβλητή**index1**. Η**angle= data.substring(0, index1);** διαβάζει τα δεδομένα από την θέση “ 0 “ μέχρι την θέση της μεταβλητής **index1** ή αλλιώς την τιμή της γωνία που στέλνεται στην σειριακή θύρα από το Arduino. Η **distance= data.substring(index1+1, data.length());** διαβάζει τα δεδομένα από την θέση index1 μέχρι το τέλος των τιμών που έχουμε ορίσει για την απόσταση.
Οι **iAngle = int(angle);** και**i Distance = int(distance);** μετατρέπουν τις μεταβλητές **String** σε**I nteger**.

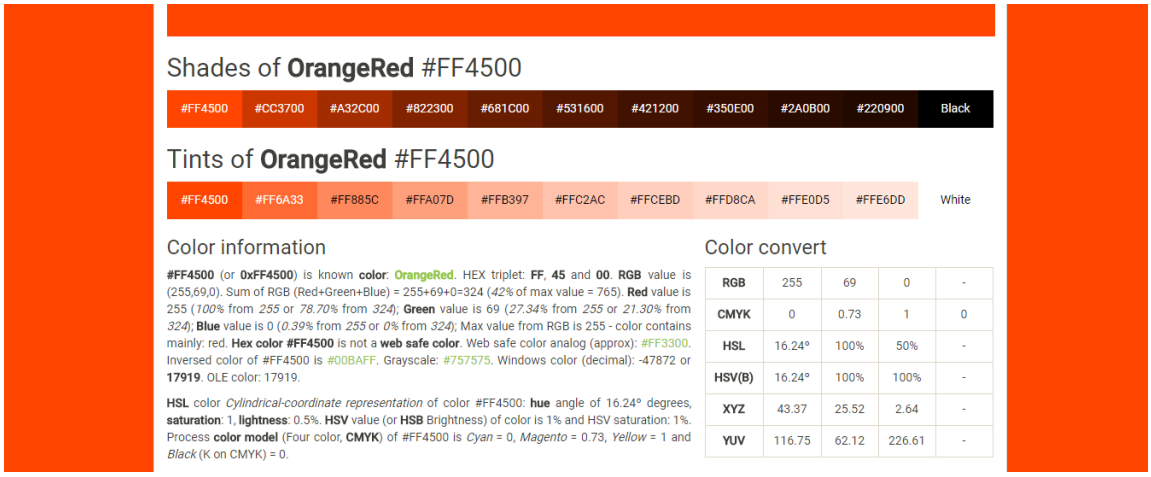
Μέσα στην **drawRadar**() θα ξεκινήσει η κίνηση των συντεταγμένων μέσα στο γραφικό μας περιβάλλονκαι θα δημιουργήσουμε τις γραμμές των τόξων.

```
arc(0,0,(width-width*0.0625),(width-width*0.0625),PI,TWO_PI);
arc(0,0,(width-width*0.27),(width-width*0.27),PI,TWO_PI);
arc(0,0,(width-width*0.479),(width-width*0.479),PI,TWO_PI);
arc(0,0,(width-width*0.687),(width-width*0.687),PI,TWO_PI);
```

Και τις γραμμές των γωνιών

```
line(-width/2,0,width/2,0);
line(0,0,(-width/2)*cos(radians(30)),(-width/2)*sin(radians(30)));
line(0,0,(-width/2)*cos(radians(60)),(-width/2)*sin(radians(60)));
line(0,0,(-width/2)*cos(radians(90)),(-width/2)*sin(radians(90)));
line(0,0,(-width/2)*cos(radians(120)),(-width/2)*sin(radians(120)));
line(0,0,(-width/2)*cos(radians(150)),(-width/2)*sin(radians(150)));
line((-width/2)*cos(radians(30)),0,width/2,0);
```

Μέσα στην **drawObject**() ,όπως και παραπάνω η **translate(width/2,height-height*0.074);** θα ξεκινήσει η κίνηση των συντεταγμένων μέσα στο γραφικό μας περιβάλλον στην επόμενη περιοχή μέτρησης. Αυτή την φορά όμως θα είναι για τα ανιχνευμένα αντικείμενα , τα οποία θα απεικονίζονται με πορτοκαλί χρώμα στην **stroke(255,69,0);**

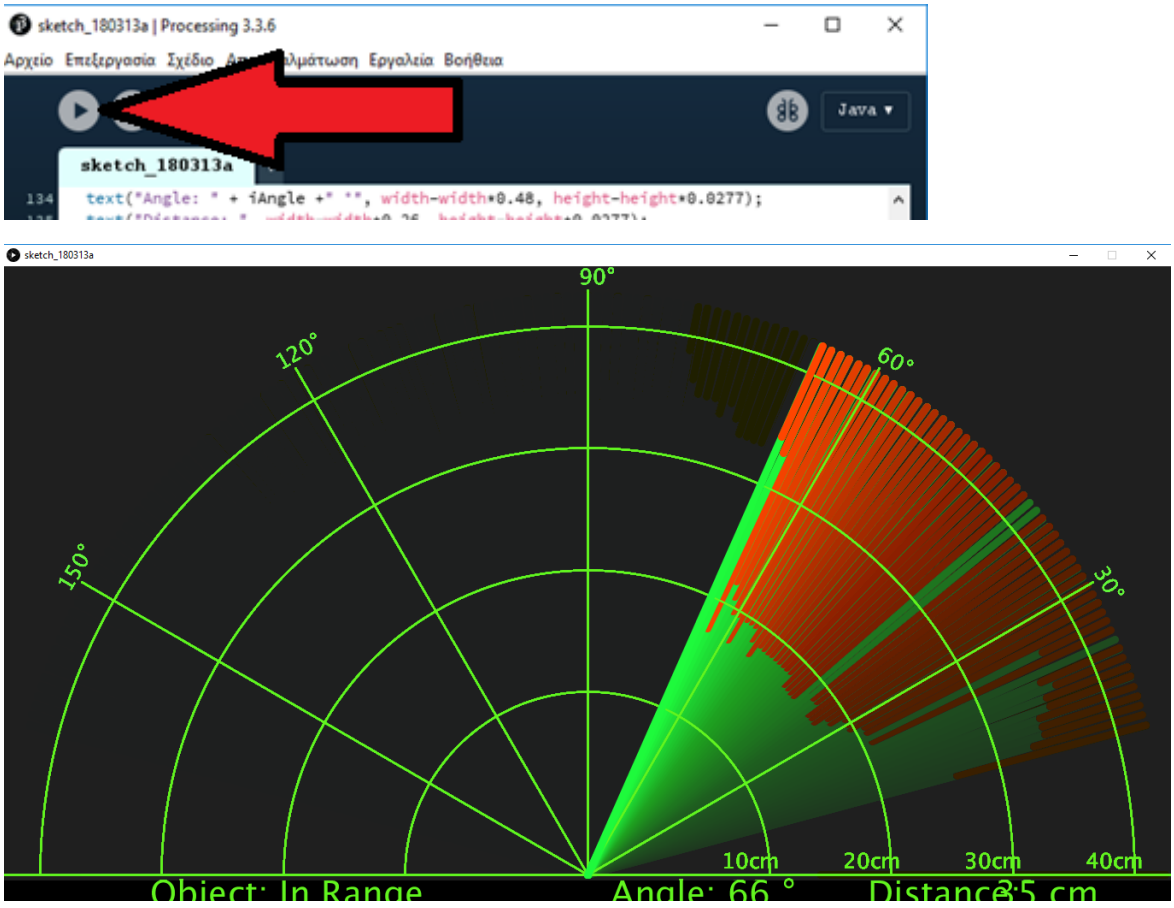


Εικόνα8.3 Χρώμα απεικόνισης ανιχνευμένου αντικειμένου

Η **pixsDistance = iDistance*((height-height*0.1666)*0.025);** μετατρέπει την απόσταση από το αντικείμενο που μετριέται από τον αισθητήρα HC-SR04 από cm σε pixels. Με την **if(iDistance<40)** περιορίζουμε την απεικόνιση των αντικειμένων έως και 40 εκατοστά από τον αισθητήρα HC-SR04 και η **line(pixsDistance*cos(radians(iAngle)),-pixsDistance*sin(radians(iAngle)),(width-width*0.505)*cos(radians(iAngle)),-(width-width*0.505)*sin(radians(iAngle)));** Δημιουργεί τα αντικείμενα στο γραφικό περιβάλλον μας σύμφωνα με την απόσταση και την γωνία που βρίσκονται από τον αισθητήρα.

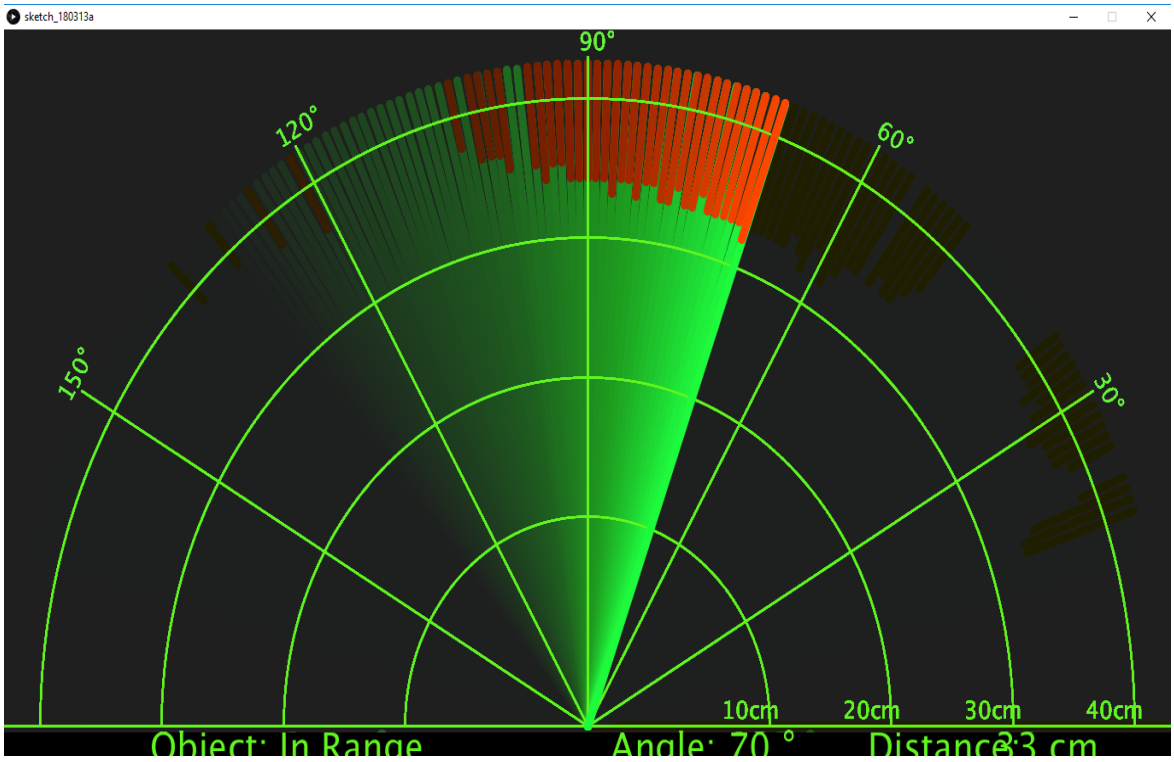
Τέλος η **drawText**() δημιουργεί το μήνυμα **OutofRange** , ή **InRange** αντίστοιχα στο κάτω αριστερά μέρος όταν κάποιο αντικείμενο βρίσκεται στην εμβέλεια του αισθητήρα η όχι , τα marks των **10,20**, **30**, **40** εκατοστών στα τόξα του γραφικού περιβάλλοντος , την γωνία και την απόσταση του αντικειμένου κλπ.

Μόλις συνδέσαμε το Arduino μας και πατήσαμε το κουμπί εκτέλεση στο ProcessingIDE το ServoMotor μας έκανε reset , γύρισε στις 15 μοίρες και άρχισε να απεικονίζει την απόσταση και τον όγκο των αντικειμένων που βρίσκονταν γύρω του.



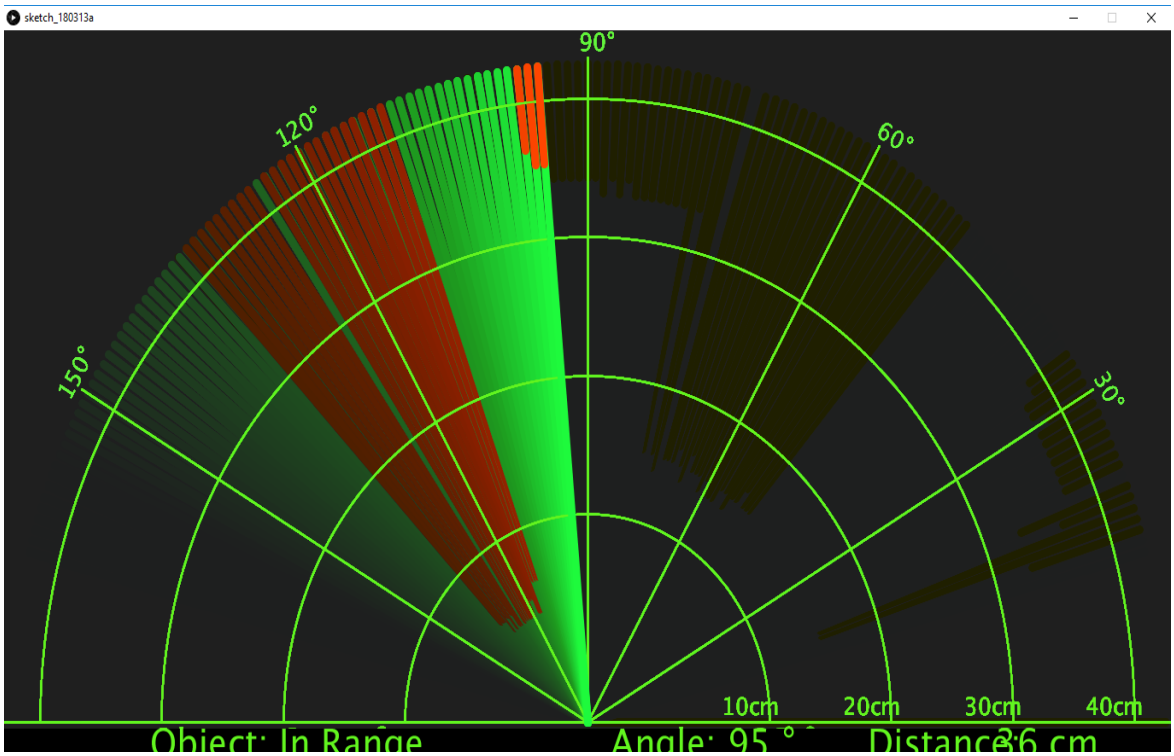
Εικόνα8.4 Γραφικό περιβάλλον Sonarσε λειτουργία

Καθώς είχα αφήσει τα ακουστικά του υπολογιστή ακριβώς μπροστά από τον αισθητήρα ο αισθητήρα τα ανίχνευσε κατευθείαν ενώ όταν τα απομάκρυνα ανίχνευσε μόνο τον τοίχο που βρισκόταν σε μεγαλύτερη απόσταση.

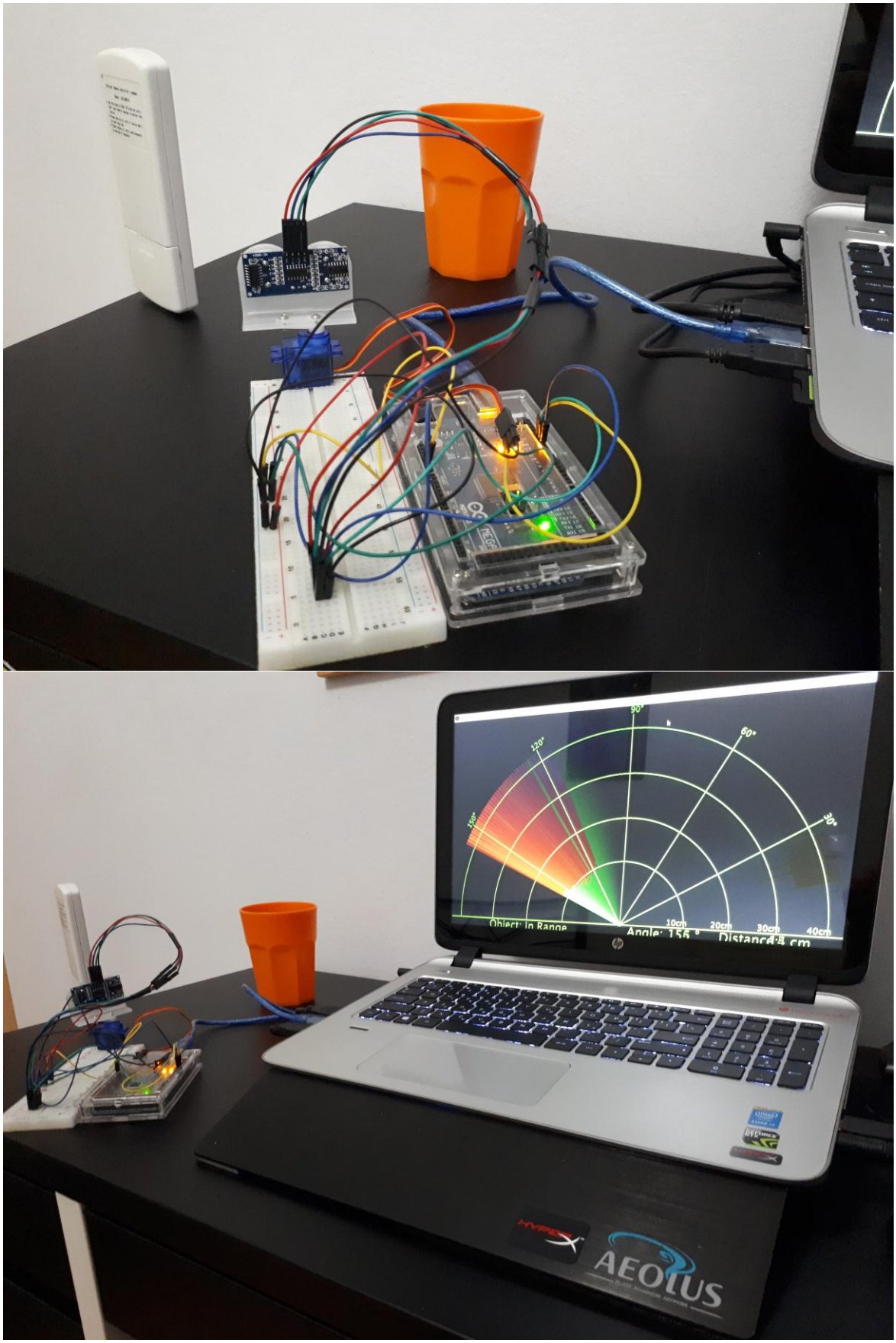


Εικόνα8.5 Γραφικό περιβάλλον Sonarσε λειτουργίαno.2

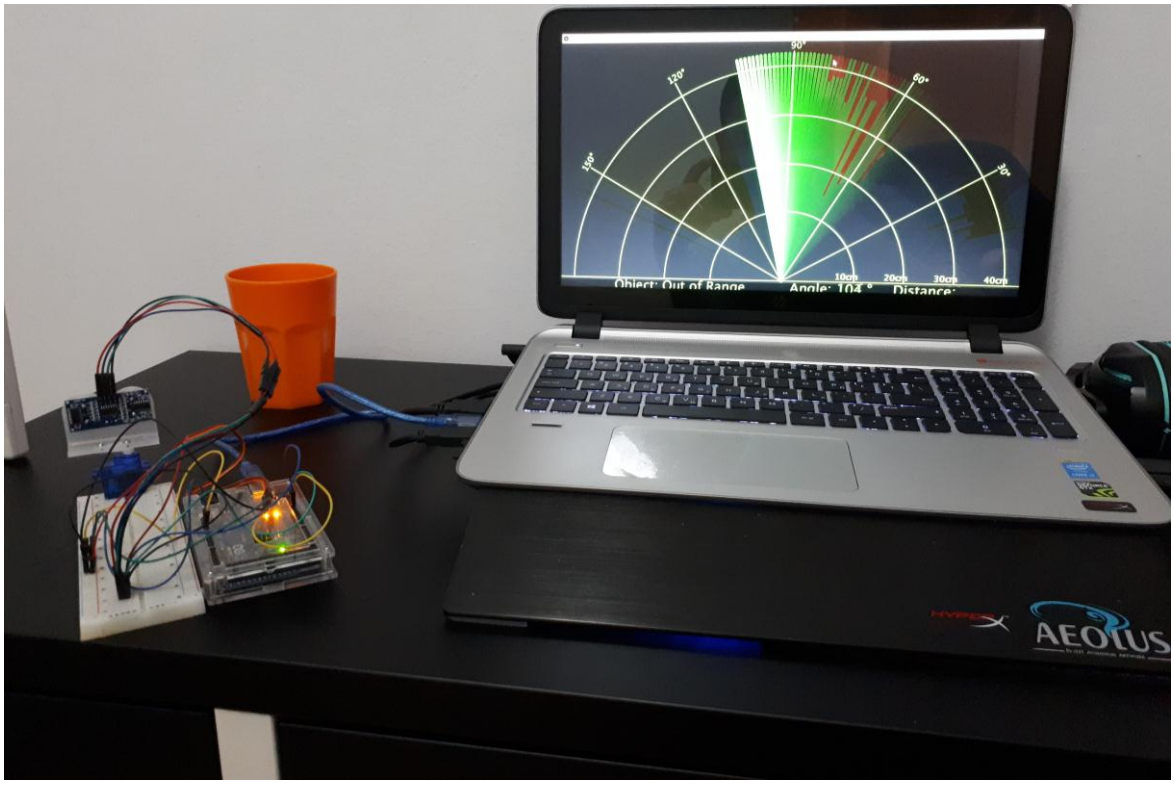
Ακολούθησαν πολλά τεστ με αλλαγή και τοποθέτηση περισσότερων αντικειμένων και το κύκλωμα του Sonar μας λειτούργησε άψογα και μπόρεσε να διακρίνει με ακρίβεια τις αλλαγές αυτές.



Εικόνα8.6 Γραφικό περιβάλλον Sonarσε λειτουργίαno.3

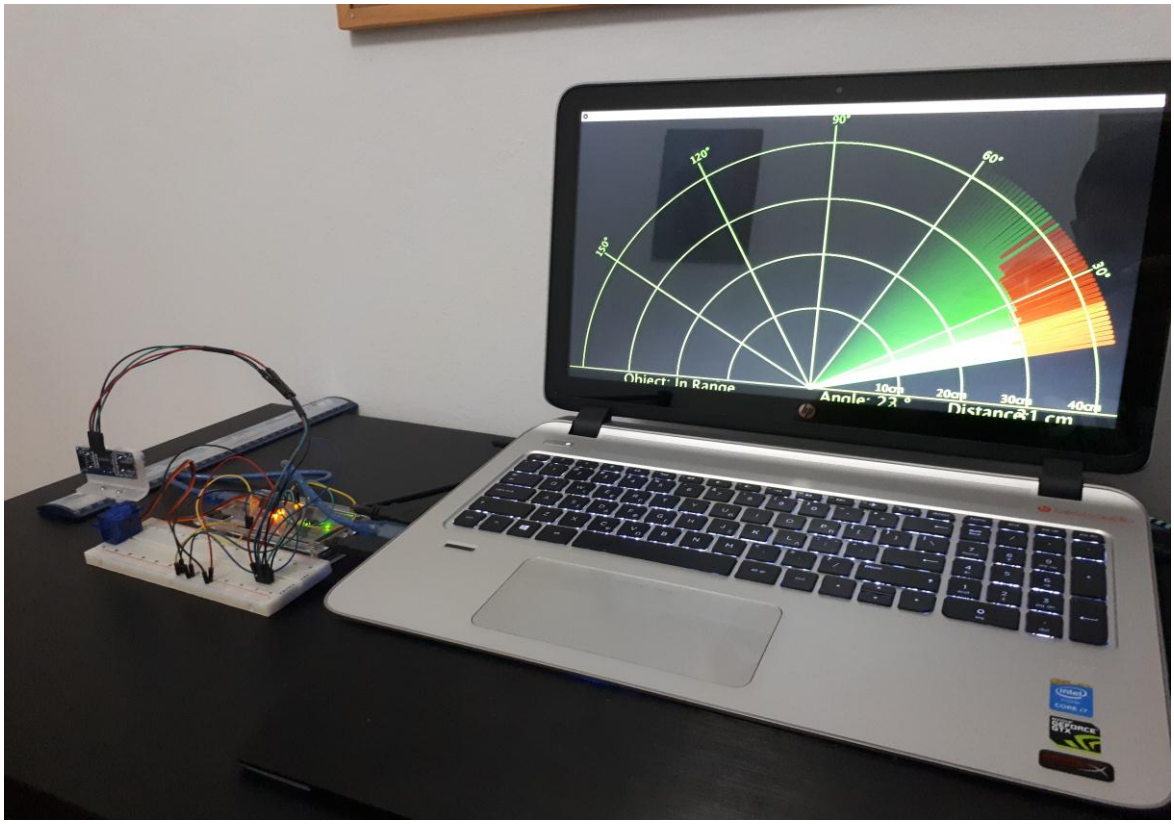


Εικόνα8.7 Γραφικό περιβάλλον και φυσικό κύκλωμα μαζί με 2 αντικείμενα



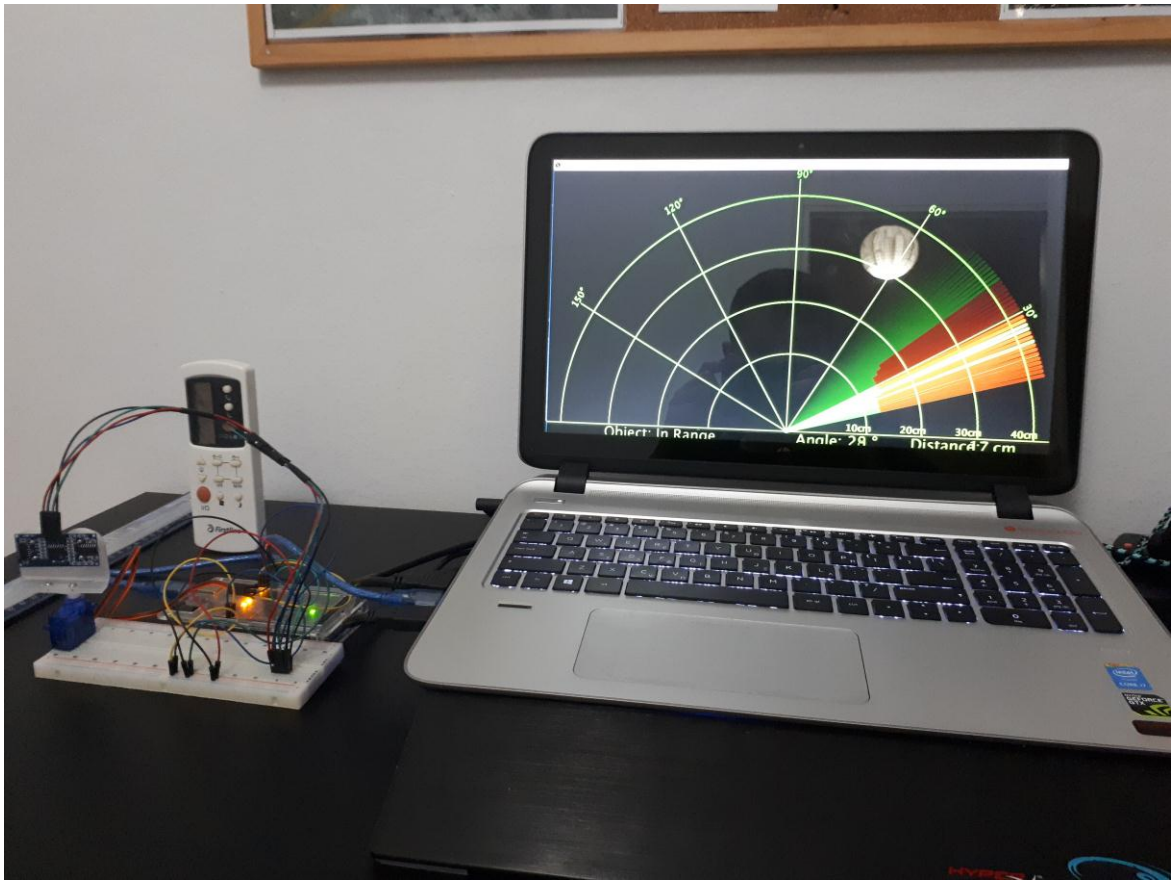
Εικόνα8.8Γραφικό περιβάλλον και φυσικό κύκλωμα μαζί με 2 αντικείμενα no.2

Καθώς ήθελα οι μετρήσεις να είναι όσο πιο ακριβείς γίνεται τοποθέτηση ενός χάρακα 30 εκατοστά από τον τοίχο. Ο αισθητήρα ανιχνεύει σωστά τον τοίχο στα 30 εκατοστά.



Εικόνα8.9 Τοποθέτηση αισθητήρα HC-SR04 και ορθή μέτρηση στο γραφικό περιβάλλον

Μετά τοποθετούμε ένα αντικείμενο στα 15 εκατοστά από τον αισθητήρα και ο εντοπισμός του αντικειμένου γίνεται πάλι στην σωστή απόσταση.



Εικόνα8.10Τοποθέτηση εμποδίου και ορθή μέτρηση στο γραφικό περιβάλλον

9. Συμπεράσματα

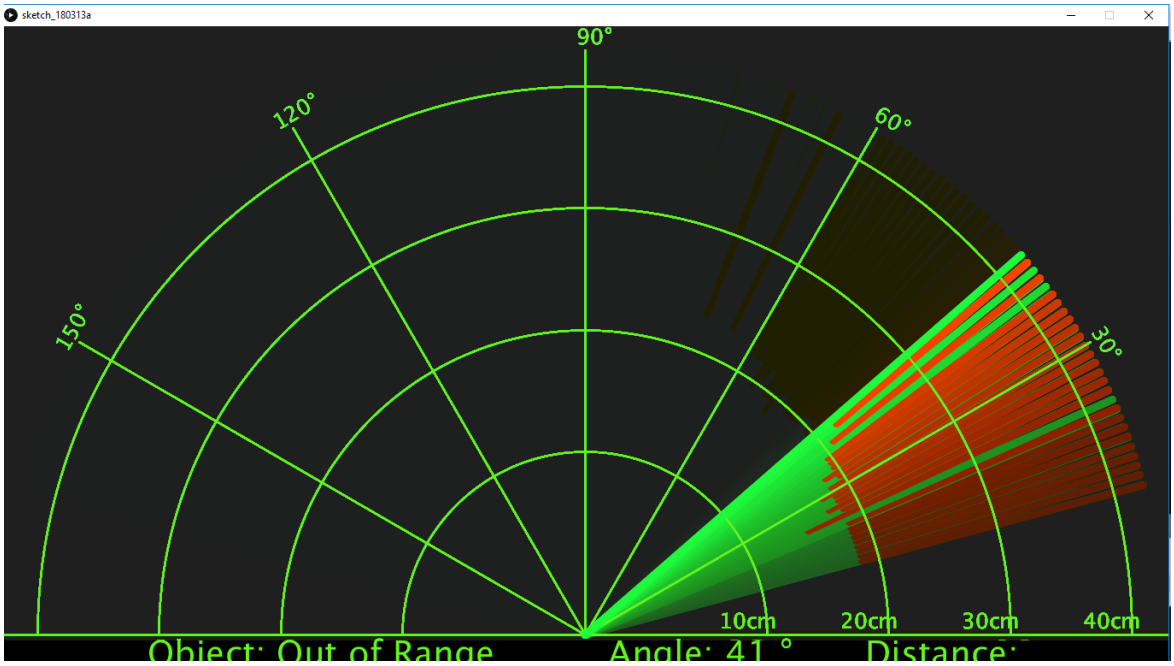
Το κύκλωμα Sonarτο οποίο δημιουργήσαμε αρχικά στο Fritzingκαι αργότερα στην φυσική του μορφή συνεργάστηκε άψογα με τον κώδικα τον οποίο φορτώσαμε στο ArduinoMega 2560 και γράφτηκε στο ArduinoSoftwareIDE. Με το γραφικό περιβάλλον που σχεδιάσαμε για τοSonarμας το οποίο γράφτηκε με Processing , με τις παρούσες ρυθμίσεις το κύκλωμα μας απεικονίζει τα αντικείμενα που βρίσκονται έως και 40 εκατοστά από τον αισθητήρα HC-SR04.

Αντικείμενα που έχουν ύψος πολύ λιγότερο από τα 6.3 εκατοστά όπου έχει τοποθετηθεί ο αισθητήρας HC-SR04 είναι αδύνατο να εντοπιστούν, ενώ με την λειτουργία του ServoMotorτο οποίο προγραμματίσαμε να εκτελεί περιστροφή από 15 μοίρες έως 165 δίνει στον αισθητήρα ο οποίον είναι τοποθετημένος πάνω στο σερβομηχανισμό του την δυνατότητα εντόπισης αντικειμένων σε φάσμα 150 μοιρών.

Η κίνηση του ServoMotorξεκινάει από τις 15 μοίρες έως τις 165 και μετά γυρνάει πίσω στις 15 όπου και επαναλαμβάνεται η διαδικασία μέχρι να διακόψουμε την σύνδεση με τον υπολογιστή. Σημεία των 10, 20, 30 και 40 εκατοστών έχουν δημιουργηθεί για την σωστή απεικόνιση του γραφικού περιβάλλοντος παρόμοιου με αυτά των Sonarόπως και τα σημεία των 30, 60, 90, 120, 150 μοιρών πάνω από τους άξονες του Sonar. Τα αντικείμενα μας παρουσιάζονται αφού εντοπιστούν με πορτοκαλί χρώμα ενώ ο κενός χώρος που δεν βρίσκεται κάποιο αντικείμενο με ανοιχτό πράσινο. Στο κάτω μέρος της Οθόνης από το Sonar Απεικονίζεται η απόσταση (**Distance**) και μετριέται σε εκατοστά ,ακριβώς δίπλα της η γωνία που βρίσκεται ο αισθητήρας (**Angle**) και μετριέται σε μοίρες.

Στο κάτω αριστερό μέρος της οθόνης έχει δημιουργηθεί μία ένδειξη η οποία αλλάζει με γνώμονα αν έχει εντοπιστεί κάποιο αντικείμενο μέσα στην εμβέλεια του HC-SR04 . Αυτή η ένδειξη προβάλλεται ως **Object**: **InRange** εάν έχει εντοπιστεί κάποιο αντικείμενο από τον HC-SR04 και **Object**: **OutofRange** εάν δεν έχει εντοπιστεί. Η λειτουργία του Sonarμας είναι άμεση, από την στιγμή που συνδέουμε το Arduinoστην USBθύρα του υπολογιστή μας ξεκινάει να ανιχνεύει και να περιστρέφεται.

Μόλις ανοίξουμε το Sketchπου έχουμε δημιουργήσει με την Processing και πατήσουμε το κουμπί **Εκτέλεση** , το κύκλωμα κάνει resetκαι ξεκινάει την λειτουργία του από τις 15 μοίρες . Εάν αφαιρέσουμε το Arduinoαπό τον υπολογιστή μας καθώς λειτουργεί το Sonarθα πρέπει να ξαναεκτελέσουμε το Sketch στην Processingγια να ξαναλειτουργήσει το περιβάλλον απεικόνισης.



Εικόνα8.11Γραφικό περιβάλλον τύπου Sonarδημιουργημένο σε Processing

10. Παραμετροποιήσεις Κώδικα

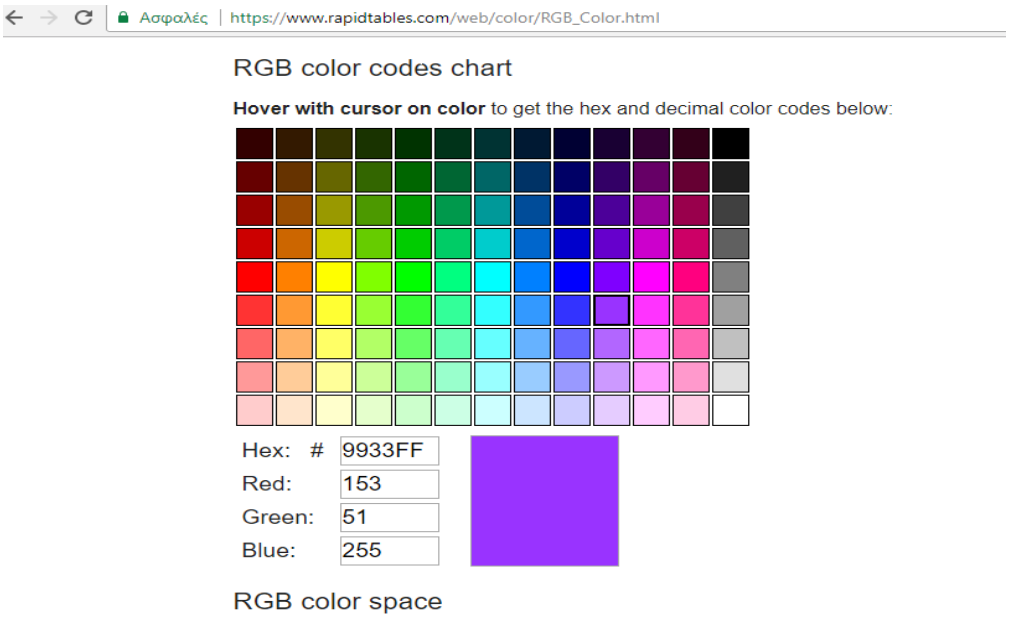
Δεκάδες είναι οι παραμετροποιήσεις που μπορούμε να κάνουμε στον κώδικα μας και αλλά θα αναφέρουμε κάποιες από τις βασικές. Αρχικά θα μπορούσαμε να επεκτείνουμε την εμβέλεια του Sonarμας μέσα από τον κώδικα καθώς ο αισθητήρα HC-SR04 έχει την δυνατότητα εντόπισης αντικειμένων σε απόσταση έως 4 μέτρα. Εμείς επιλέξαμε μια πολύ κοντινή απόσταση , τα 40 εκατοστά για να έχουμε όσο το δυνατόν καλύτερη ακρίβεια μπορούμε στις μετρήσεις της απόστασης. Στην γραμμή **if(iDistance<40){** θα μπορούσαμε να αλλάξουμε τα εκατοστά αλλάζοντας την τιμή που γράφει το **iDistance** και ο αισθητήρα μας θα μπορούσε να μετρήσει μεγαλύτερη ή μικρότερη απόσταση αντίστοιχα. Πρόβλημα με το γραφικό περιβάλλον θα αντιμετωπίζαμε ένα αλλάζαμε την απόσταση σε μία μεγαλύτερη καθώς θα έπρεπε να σχεδιάσουμε επιπλέον τόξα και να προεκτείνουμε τους άξονες καθώς και να προσθέσουμε καινούριες ενδείξεις της απόστασης.

Ακόμη θα μπορούσαμε να παραμετροποιήσουμε την ένδειξη **Distance**στο κάτω μέρος του γραφικού περιβάλλοντος, αντί να μετράει την απόσταση σε εκατοστά να την μετράει σε Inches. Πάλι θα αντιμετωπίζαμε πρόβλημα με το γραφικό περιβάλλον καθώς θα έπρεπε να αλλάξουμε τις ενδείξεις των εκατοστών σε ενδείξεις για Inches.

Ηένδειξη**Object : (InRange – OutofRange)** είναι τελείως περιττή και μια παραμετροποίηση που θα μπορούσαμε να κάνουμε είναι να την αφαιρέσουμε διαγράφοντας τις γραμμές του κώδικα που την δημιουργήσαν.

```
if(iDistance>40) {
  noObject = "Out of Range";
}
  else {
  noObject = "In Range";
  }
```

Ακόμη, το χρώμα απεικόνισης των ενδείξεων , των αξόνων, των τόξων, του κενού χώρου θα μπορούσαμε να το αλλάξουμε σε οποιοδήποτε άλλο χρώμα θέλαμε από την γραμμή **stroke(255,69,0);** Μέσα στην **void**του κάθε αντικειμένου. Το χρώμα που θέλουμε θα μπορούσαμε να το βρούμε στο διαδίκτυο μέσα σε οποιοδήποτε RGBColorchartαντιγράφοντας τις τιμές (R, G, B) και τοποθετώντας τες πάνω από τις ήδη υπάρχουσες.



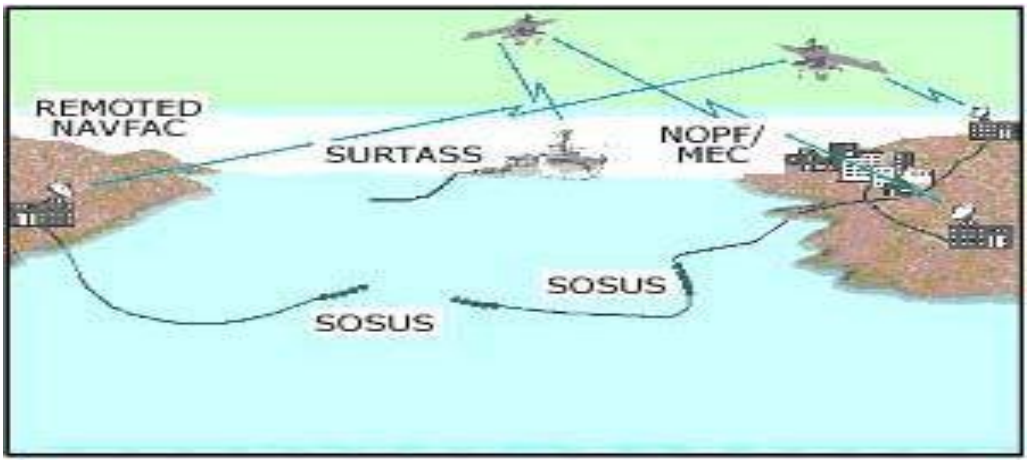
Εικόνα 8.12Online RGB Color Code Chart

11. Εφαρμογές και εξελικτική διαδικασία

Η χρήση του σόναρ που ουσιαστικά έχει συμπληρώσει έναν αιώνα και πλέον πρακτικής εφαρμογής δεν περιορίζεται σε Στρατιωτική χρήση. Πρόσφατο παράδειγμα χρήσης των SONAR είναι η περίπτωση της προσπάθειας εντοπισμού του Boeing 777 της Malaysian Airlines (που δυστυχώς μέχρι σήμερα δεν έχει αποδώσει).Για την ακρίβεια ένα πλοίο που είχε εγκατεστημένο ένα παθητικό SONAR προσπαθούσε να «ακούσει» εκπομπές που πιθανόν να προέρχονταν από το μαύρο κουτί του αεροπλάνου και μάλιστα μια εκπομπή που έλαβε ήταν αυτή στη φασματική συχνότητα των 37.5 kHz.

Επίσης η χρήση του SONAR για την χαρτογράφηση της θαλάσσιας περιοχής στα ανοικτά της Νήσου Σκύρου, όπου την 12^{η} Απριλίου 2018 κατέπεσε αεροσκάφος της Π.Α με αποτέλεσμα τον θάνατο του πιλότου, είναι ένα πολύ πρόσφατο δείγμα της σπουδαιότητας της συσκευής ‘όσον αφορά την διερεύνηση αεροναυτικών ατυχημάτων.

Φυσικά οι στρατιωτικές εφαρμογές των SONAR όπου τα πλοία/ υποβρύχια προσπαθούν να εντοπίσουν εχθρικά υποβρύχια χωρίς όμως να προδώσουν τη θέση τους, η πλοήγηση υποβρυχίων στρατιωτικών και μη και η επικοινωνία μεταξύ αυτών, κατέχουν την μερίδα του λέοντος όσον αφορά την χρήση των εξελιγμένων πλέον αυτών συσκευών. Για να “ακούνε” τον ωκεανό υπάρχει το SOSUS (SOund Surveilance System) που αποτελείται από συστοιχίες υδροφώνων εγκατεστημένων στο βυθό των ωκεανών, τα οποία μέσω υποβρύχιων καλωδίων συνδέονται με εγκαταστάσεις στη στεριά. Βρίσκονταιστη περιοχή SOFAR ή σε ακόμα μεγαλύτερο βάθος, είναι παθητικά συστήματα και έχουν τη δυνατότητα να ανιχνεύουν και ασθενή σήματα καθώς βρίσκονται σε ήσυχο περιβάλλον.



Άλλες εφαρμογές, μη στρατιωτικές είναι η χαρτογράφηση του βυθού και ο εντοπισμός οικογενειών ψαριών , γεγονός που απαιτεί μεγάλη διακριτική ικανότητα με δεδομένο ότι το 94% της ζωής σε αυτό τον πλανήτη βρίσκεται κάτω από την θάλασσα.

Υπάρχει επιστημονικός κλάδος που ονομάζεται υδρογραφία, που μελετά το βυθό και δημιουργεί διαγράμματα για χαρακτηριστικά του. Οι επιστήμονες υπολογίζουν

το βάθος του ωκεανού, και αναζητούν υφάλους, πέτρες και ναυάγια που θα μπορούσαν να βλάψουν την πλοήγηση των υποβρυχίων. Τα δεδομένα χαρτογραφούνται και προκύπτουν οι ναυτικοί χάρτες-διαγράμματα.

Πλέον , υπάρχει και το GIS (Geographic Information System), στο οποίο αν δοθούν αρκετά δεδομένα μπορεί να κατασκευάσει έναν εικονικό ωκεανό μέσα στον υπολογιστή. Οι συχνότητες λειτουργίας των σόναρ κυμαίνεται ανάλογα με την εφαρμογή από πολύ χαμηλές- infrasonics- κάτω από 20Hz ως πολύ υψηλές – ultrasonics -20kHz ως κάποια GHz. Οι συχνότητες λειτουργίας θεωρούνται χαμηλές όταν είναι 100-500Hz, και χρησιμοποιούνται για εντοπισμό μεγάλων αντικειμένων από απόσταση αλλά με χαμηλή ανάλυση. Παράδειγμα λειτουργίας στις συχνότητες αυτές αποτελεί το Low Frequency Active SONAR.

Το αμερικάνικο Ναυτικό έχει στόχο να τοποθετήσει LFA στο 80% του ωκεανού για τον εντοπισμό “ήσυχων” υποβρυχίων, τα οποία θα σύρονται πίσω από πλοία. Χαρακτηρίζονται ως μεσαίες όταν είναι 1kHz με 10kHz- συνήθως σε στρατιωτικές εφαρμογές για εντοπισμό υποβρυχίων, και πάνω από 20 kHz που χρησιμοποιούνται για χαρτογράφηση του βυθού σε σχετικά ρηχές θάλασσες από multibeam SONAR ή σε μεγάλο βάθος από μεταβλητού βάθους SONAR με υψηλή ανάλυση. Χαρακτηριστικές υψηλές συχνότητες είναι τα 120kHz , τα 200kHz και τα 455kHz. Σύγχρονες ακουστικές πηγές παράγουν κύματα και 1 GHz.

Το μέλλον του Sonar σίγουρα δεν είναι αόρατο. Η Αγγλική Κυβέρνηση έχει επίσης πρόσφατα αναπτύξει μια εναλλακτική μελλοντική λύση ανίχνευσης για εφαρμογές SONAR, η οποία χρησιμοποιεί ακουστικά επαγόμενες αλλαγές ηλεκτρικής αντίστασης στο θαλάσσιο νερό που περιβάλλει ένα σκάφος για την ανίχνευση ήχων. Το σύστημα μέτρησης ηλεκτρικής αντίστασης για ανίχνευση ακουστικών σημάτων (MIDAS) χρησιμοποιεί απλούς, χαμηλής ισχύος, συμπαγείς αισθητήρες που παρέχουν εγγυημένα αποτελέσματα με χαμηλό κόστος και μεγάλη διάρκεια ζωής.

Αν σκεφτούμε ότι το 73% του πλανήτη καλύπτεται απόθάλασσα και μεγάλο μέρος των ωκεανών (95%)παραμένει ανεξερεύνητο ή έστω όχι επαρκώς εξερευνημένο, οι εφαρμογές του SONAR αποτελούν ένα αναντικατάστατο εργαλείο για την εξερεύνηση των ωκεανών και σε συνδυασμό με άλλες συσκευές το κλειδί για την εκμετάλλευση των πλουτοπαραγωγικών πηγών που κρύβονται στα βάθη τους.

12. Πηγές - Βιβλιογραφία

Sonar
https://en.wikipedia.org/wiki/Sonar
https://www.uio.no/studier/emner/matnat/ifi/INF-GEO4310/h11/undervisningsmateriale/sonar_presentation_2011_compressed.pdf
https://www.nationalgeographic.org/photo/fessenden-oscillator/
https://asa.scitation.org/doi/abs/10.1121/1.409629
https://en.wikipedia.org/wiki/Fessenden_oscillator
http://www.physics-and-radio-electronics.com/blog/sonar-or-sound-navigation-and-ranging/

Arduino
https://el.wikipedia.org/wiki/Arduino
https://en.wikipedia.org/wiki/Arduino
https://www.arduino.cc/en/Main/Boards
https://www.microchip.com/wwwproducts/en/ATmega168
https://www.arduino.cc/en/uploads/Main/arduino-mega2560_R3-sch.pdf
https://www.arduino.cc/en/uploads/Main/arduino-mega2560-schematic.pdf
https://store.arduino.cc/usa/arduino-mega-2560-rev3
https://www.arduino.cc/en/Hacking/PinMapping2560
https://www.arrow.com/en/research-and-events/articles/arduino-mega-2560-overview

Αισθητήρας HC-SR04
http://osoyoo.com/2017/07/23/arduino-lesson-ultrasonic-sensor-hc-sr04/
https://cdn.sparkfun.com/datasheets/Sensors/Proximity/HCSR04.pdf
https://www.mpja.com/download/hc-sr04_ultrasonic_module_user_guidejohn.pdf
https://elecfreaks.com/estore/download/EF03085-HC-SR04_Ultrasonic_Module_User_Guide.pdf

Servo Motor (Σερβοκινητήρας)
http://www.acdcshop.gr/content/POLOLU-2820.pdf
https://www.pololu.com/product/2820
https://arduinobots.wordpress.com
http://www.metadosi-ischios.gr/article.php?ID=83

Υπόλοιποι σύνδεσμοι
https://playground.arduino.cc/Interfacing/Processing
https://www.arduino.cc/en/Guide/Environment
http://brain.ee.auth.gr/dokuwiki/doku.php?id=sonar:sonar
http://www.hobbyist.co.nz/?q=putting-the-sonar-together
https://howtomechatronics.com/projects/arduino-radar-project/
http://www.hobbyist.co.nz/?q=arduino-servo-sonar
http://www.hobbyist.co.nz/?q=setting-up-processing